



УНИВЕРЗИТЕТ У КРАГУЈЕВЦУ
ФАКУЛТЕТ ИНЖЕЊЕРСКИХ НАУКА

Богдан Ж. Милићевић

**СУРОГАТ МОДЕЛИ МИШИЋА ЗАСНОВАНИ НА
ВЕШТАЧКИМ НЕУРОНСКИМ МРЕЖАМА СА
ПРИМЕНОМ У АНАЛИЗИ МЕТОДОМ КОНАЧНИХ
ЕЛЕМЕНАТА**

докторска дисертација

Крагујевац, 2022



UNIVERSITY OF KRAGUJEVAC
FACULTY OF ENGINEERING

Bogdan Ž. Milićević

**SURROGATE MUSCLE MODELS BASED ON ARTIFICIAL
NEURAL NETWORKS WITH APPLICATIONS IN FINITE
ELEMENT ANALYSIS**

Doctoral Dissertation

Kragujevac, 2022

Аутор
Име и презиме: Богдан Милићевић
Датум и место рођења: 20.08.1992. године, Крагујевац, Република Србија
Садашње запослење: Истраживач-сарадник, Факултет инжењерских наука Универзитета у Крагујевцу
Докторска дисертација
Наслов: Сурогат модели мишића засновани на вештачким неуронским мрежама са применом у анализи методом коначних елемената
Број страница: 118
Број слика: 97
Број библиографских података: 71
Установа и место где је рад израђен: Факултет инжењерских наука Универзитета у Крагујевцу, Универзитет у Крагујевцу
Научна област (УДК): 004 (Рачунарска наука и технологија. Рачунарство)
Ментор: др Ненад Филиповић, редовни професор, Факултет инжењерских наука, Универзитета у Крагујевцу
Оцена и одбрана
Датум пријаве теме: 10.12.2019.
Број одлуке и датум прихватања теме докторске/уметничке дисертације: IV-04-317/07 10.06.2020.
Комисија за оцену научне заснованости теме и испуњености услова кандидата:
<ol style="list-style-type: none"> 1. Др Ненад Филиповић, редовни професор, Факултет инжењерских наука, Универзитет у Крагујевцу 2. Др Миљан Милошевић, ванредни професор, Универзитет Метрополитан, Универзитет у Београду 3. Др Весна Ранковић, редовни професор, Факултет инжењерских наука, Универзитет у Крагујевцу 4. Др Милош Ивановић, ванредни професор, Природно-математички факултет, Универзитет у Крагујевцу 5. Др Бобан Стојановић, редовни професор, Природно-математички факултет, Универзитет у Крагујевцу
Комисија за оцену и одбрану докторске/уметничке дисертације:
<ol style="list-style-type: none"> 1. Др Миљан Милошевић, ванредни професор, Универзитет Метрополитан, Универзитет у Београду 2. Др Весна Ранковић, редовни професор, Факултет инжењерских наука, Универзитет у Крагујевцу 3. Др Милош Ивановић, ванредни професор, Природно-математички факултет, Универзитет у Крагујевцу 4. Др Бобан Стојановић, редовни професор, Природно-математички факултет, Универзитет у Крагујевцу 5. Др Дубравко Ђулибрк, редовни професор, Факултет техничких наука, Универзитет у Новом Саду
Датум одбране дисертације:

Захвалнице

Најпре захвалност дугујем ментору проф. др Ненаду Филиповићу, редовном професору Факултета инжењерских наука у Крагујевцу, на пруженој прилици да се бавим истраживањима у области биоинжењеринга и рачунарских наука. Хвала на усмеравању током докторских студија и на прилици да будем део пројеката Horizont2020 (SILICOFCM (бр. 777204), SGABU (бр. 952603)).

У току докторских студија, имао сам велику част да сарађујем са академиком проф. др Милошем Којићем, од кога сам научио пуно и коме сам захвалан на корисним саветима.

Велику захвалност дугујем ванредном професору Универзитета Метрополитан, др Миљану Милошевићу, на сарадњи и уложеном времену и труду у току мог усавршавања у области биоинжењеринга, рачунарских симулација и моделирања. Такође, велику захвалност дугујем професорима Природно-математичког факултета у Крагујевцу, ванредном професору др Милошу Ивановићу и редовном професору др Бобану Стојановићу, на пруженом образовању на основним и мастер студијама, а затим и на разним предлозима у току мојих истраживања у области сурогат моделирања и вештачке интелигенције. Хвала др Весни Ранковић, редовном професору Факултета инжењерских наука у Крагујевцу, на пријатној сарадњи и корисним саветима и сугестијама у току докторских студија. Хвала, редовном професору Факултета техничких наука у Новом Саду, др Дубравку Ђулибрк на корисним предлозима.

Хвала колегама Владимиру Симићу, Жики Милановићу и Марку Антонијевићу, као и колегиницама Јелени Ђоровић и Светлани Јеремић из Центра за биоинжењеринг, на веома пријатном дружењу и подршци. Хвала Лазару Васовићу на сарадњи у истраживањима у области сурогат моделирања. Захвалност на сарадњи такође дугујем професорима, колегама и колегиницама са Факултета инжењерских наука у Крагујевцу.

На крају, највећу захвалност дугујем својој породици, мајци Ружици, оцу Животију и сестри Сари, на безусловној подршци и љубави.

*Крагујевац, 2022. године
Богдан Милићевић*

Сажетак

Биофизички модели мишића, засновани на физиолошким принципима функционисања мишића, се могу користити да одреде механички одзив мишића прецизније него феноменолошки модели, који су засновани на експерименталним мерењима. Међутим, за разлику од феноменолошких, биофизички модели мишића су рачунски веома захтевни, што отежава њихову употребу у вишескалним симулацијама. Типичан пример биофизичког модела мишића је Хакслијев модел. У овој дисертацији, да би се омогућила ефикаснија употреба биофизичких модела, креирани су сурогат модели засновани на вештачким неуронским мрежама такви да они имитирају оригинални Хакслијев модел, али користе мању количину меморије и других рачунарских ресурса. Најбољи резултати су постигнути затвореним рекурентним јединицама, које су дале најтачније напоне у мишићу од свих конструисаних мрежа. У различитим нумеричким експериментима је показано да су предвиђени напони и тренутна крутост скоро потпуно исти као оригинални. Показано је да је конструисани сурогат модел за ред величине бржи од Хакслијевог модела решаваног класичним нумеричким поступком и да троши мању количину меморије. Поред тога, у овој дисертацији су приказане неуронске мреже подржане физичким законима, које су обучаване тако да апроксимирају решење Хакслијеве једначине за мишићну контракцију. Показано је да вишеслојни перцептрон, подржан физичким законима, боље генерализује понашање мишића него стандардни вишеслојни перцептрон.

У овој дисертацији су представљене процедуре за креирање сурогат модела мишића, заједно са процедурама за интеграцију сурогат модела у софтверски оквир за анализу методом коначних елемената. Да би потенцијал сурогат модела за коришћење у захтевним вишескалним симулацијама био демонстриран у пуном обиму, симулиран је срчани циклус леве коморе, што би било значајно теже урадити са оригиналним Хакслијевим моделом.

Кључне речи: вештачке неуронске мреже, рекурентне неуронске мреже, неуронске мреже подржане физичким законима, сурогат модели, Хакслијев модел мишића, метод коначних елемената, модел леве коморе

Abstract

Biophysical muscle models, which are based on the underlying physiology of the muscles, can evaluate the mechanical response of the muscles more accurately than phenomenological muscle models, which are based on experimental measurements. On the other hand, biophysical muscle models are much more computationally intensive. Biophysical muscle models are often called Huxley-type muscle models. In this dissertation, to enable the efficient use of Huxley-type muscle models in multi-scale simulations of the cardiac cycle, surrogate models were created such that they mimic the original Huxley muscle model but use less memory and processing power. The best results were achieved with the gated recurrent units, which produced the most accurate stresses. Stresses and instantaneous stiffnesses produced by the surrogate model were almost indistinguishable from the original values. The constructed surrogate model was an order of magnitude faster than Huxley's muscle model and used less memory. Additionally, in this dissertation, physics-informed neural networks were trained to approximately solve Huxley's muscle contraction equation. It was shown that the generalization of the physics-informed multilayer perceptron is greater than that of the ordinary multilayer perceptron.

The procedures for the creation of the surrogate muscle models were introduced in this dissertation, along with the procedures for the integration of the surrogate models into a finite element analysis framework. To show the potential of the surrogate models in larger-scale simulations, the cardiac cycle of the left ventricle model was simulated, which would be much higher to do with the original Huxley's muscle model.

Keywords: artificial neural networks, recurrent neural networks, physics-informed neural networks, surrogate models, Huxley muscle model, finite element method, left ventricle model

Садржај

1 Увод	1
1.1 Повезаност дисертације са досадашњим истраживањима.....	1
1.2 Предмет и допринос дисертације	2
1.3 Основне хипотезе дисертације	3
1.4 Преглед садржаја дисертације.....	4
2 Феноменолошки и биофизички модели мишића	5
2.1 Материјални модели мишића.....	9
3 Нумеричке методе за моделирање механичког одзива мишића	12
3.1 Метод карактеристика	12
3.1.1 Итеративни поступак.....	14
3.2 Метод коначних елемената	15
3.2.1 Нелинеарна анализа у механици солида	16
4 Суругат моделирање	21
4.1 Препроцесирање података	22
5 Вештачке неуронске мреже	24
5.1 Перцептрон	24
5.2 Функције трансфера	25
5.3 Вишеслојни перцептрон.....	27
5.3.1 Пропагација унапред	29
5.3.2 Пропагација уназад	30
5.3.3 Иницијализација параметара.....	32
5.4 Проблем генерализације	33
5.4.1 Типови грешака.....	35
5.4.2 Рано заустављање.....	35
5.4.3 Смањивање тежинских коефицијената.....	36
5.4.4 Регулација изостављањем неурона	36
5.5 Функције губитка	37
5.5.1 Функције губитка за проблеме регресије.....	37
5.6 Евалуација модела.....	38
5.6.1 Функције за евалуацију регресионих проблема.....	38
5.7 Алгоритми за оптимизацију.....	39
5.7.1 Градијентни спуст и његове варијације	41
5.7.2 Момент у алгоритмима за оптимизацију	43
5.8 Неуронске мреже подржане физичким законима	45
5.9 Рекурентне неуронске мреже	49

5.9.1	Одсецање градијената.....	52
5.9.2	Пропагација уназад кроз време.....	53
5.10	Унапређене верзије рекурентних неуронских мрежа.....	56
5.10.1	Затворене рекурентне јединице.....	56
5.10.2	Јединица са дуготрајним-краткотрајним памћењем.....	58
5.10.3	Угњеждена јединица са дуготрајном-краткотрајном меморијом.....	60
5.10.4	Дубоке рекурентне неуронске мреже.....	61
5.10.5	Бидирекционе рекурентне мреже.....	62
5.11	Конволуционе неуронске мреже.....	63
5.11.1	Инваријантност код конволуционих мрежа.....	63
5.11.2	Локалност код конволуционих мрежа.....	64
5.11.3	Операција конволуције.....	64
5.11.4	Конволуциони слојеви.....	67
5.11.5	Резидуалне мреже.....	69
5.12	Конволуционе неуронске мреже за процесирање секвенцијалних података.....	70
6	Креирање сурогат модела.....	74
6.1	Креирање сурогат модела заснованог на неуронским мрежама за анализу временских серија.....	74
6.2	Креирање сурогат модела заснованог на неуронским мрежама подржаним Хакслијевом једначином.....	77
7	Примена сурогат модела у анализи методом коначних елемената.....	79
8	Резултати и дискусија.....	82
8.1	Анализа прикупљених података и механизма обуке неуронске мреже.....	82
8.2	Поређење обучених неуронских мрежа за анализу временских серија.....	84
8.3	Дијаграми напона у току времена са GRU неуронском мрежом и оригиналним Хакслијевим моделом.....	86
8.4	Убрзање нумеричких симулација добијено сурогат моделирањем и потрошња меморије.....	90
8.5	Неуронске мреже подржане Хакслијевом једначином за мишићну контракцију.....	91
8.6	Параметарски модел леве коморе.....	97
8.7	Модел леве коморе генерисан на основу ехокардиографских снимака.....	100
9	Закључна разматрања.....	105
	Литература.....	107
	Додатак А.....	114

1 Увод

Кардиомиопатија означава групу болести које захватају срчани мишић. Промене на срчаном мишићу, миокарду, су обично споре, често без симптома, теку прогресивно и временом доводе до тешких последица. Узрок фамилијарне кардиомиопатије су варијације у људским генима које доводе до мутација у протеинима [1]. У питању су гени који синтетишу мишићне протеине. Генетске мутације могу довести до измењене кинетике попречних мостова и до измењених контрактилних карактеристика мишићних ћелија, па су овакве мутације одговорне за значајан проценат срчаних поремећаја [1]. Да би фамилијарна кардиомиопатија била боље проучавана, евалуацијом различитих реалних и фиктивних сценарија, и да би се лечење пацијената побољшало, треба направити рачунарске моделе.

Приликом креирања рачунарских модела, формира се математички модел система, а затим се користе нумерички поступци, пошто парцијалне диференцијалне једначине, које описују систем од интереса, најчешће није могуће решити аналитичким путем. Једна од нумеричких метода за решавање парцијалних диференцијалних једначина јесте метод коначних елемената. У току анализе методом коначних елемената, модел од интереса се моделира мрежом састављеном од једноставних геометријских форми, које називамо коначним елементима. У свакој интеграционој тачки коначних елемената потребно је одредити материјалне карактеристике, за шта се користе одговарајући материјални модели. Мишиће можемо посматрати као сложен материјал, који се контрахује услед мишићне активације, и чије је понашање условљено претходним стањем и оптерећењем. Материјални модели мишића се могу поделити на феноменолошке и биофизичке. Феноменолошки модели су засновани на експериментима, конзистентни су са фундаменталном теоријом функционисања мишића, али не потичу директно из ње. Насупрот томе, биофизички модели су засновани на физиолошком понашању мишића. Типичан пример феноменолошког модела је Хиллов модел, а типичан пример биофизичког модела је Хакслијев модел. Да би подаци из генетске анализе били повезани са њиховим последицама на понашање срца потребно је користити биофизичке материјалне моделе мишића. Ови модели мишића су погодни за моделирање неуниформних и променљивих контракција, али нису погодни за употребу у пракси, јер су рачунски неефикасни. Креирање сурогат модела, који опонашају биофизичке моделе али су рачунски ефикаснији од њих, би омогућило коришћење ових модела у сложеним рачунарским симулацијама које опонашају срчану контракцију. Циљ ове дисертације јесте креирање сурогат модела Хакслијевог модела мишића за унапред дефинисан скуп параметара, при чему се варирају мишићна активација и екстерно оптерећење.

1.1 Повезаност дисертације са досадашњим истраживањима

Вишескалне симулације са методом коначних елемената на макронивоу и Хакслијевим моделом на микронивоу су представљене у радовима [2,3]. Да би се анализирано понашање мишића, биофизички процеси се моделирају на више временских и просторних скала. Механика мишића на макронивоу је моделирана коначним елементима, а материјалне карактеристике на микроскопској скали су дефинисане Хакслијевим моделом мишића [2,3]. У току симулације, Хакслијев модел се користи да би се, на основу мишићне активације, стреча и других материјалних карактеристика и параметара, срачунали напон и извод напона односно тренутна крутост. Овакве вишескалне симулације могу бити веома рачунски захтевне. Најзахтевнији део симулација су прорачуни који се одвијају на микро-нивоу. Један корак овакве

симулације са само хиљаду коначних елемената траје више сати. Да би се овакве симулације убрзале коришћена је хибридна *MPI-GPU* паралелизација у циљу симуирања 2Д модела језика [4,5]. Убрзање добијено паралелизацијом може бити значајно, али захтева приступ рачунарским кластерима. На основу радова [4,5] очигледно је да чак и са рачунарским кластером овакве симулације захтевају пуно рачунског времена. У овој дисертацији, да би се смањили рачунски захтеви, креиран је рачунски ефикаснији сурогат модел који мења оригинални Хакслијев модел. Предност сурогат модела је велико убрзање које може бити достигнуто употребом и само једног процесора, док је мана губитак на прецизности модела.

Методe машинског учења постају све популарније у науци. Вишеслојни перцептрон је коришћен за креирање сурогат модела за композитне материјале са прогресивним оштећењем у раду [6]. Сурогат модел за Перзина виско-пластични нелинеарни материјални модел на микронивоу је представљен у раду [7], где су коришћене рекурентне јединице са дуготрајном-краткотрајном меморијом за предвиђање напона на основу стречева добијених на макронивоу. У оба рада [6,7], постигнута је задовољавајућа сличност између оригиналног и сурогат модела. Вишескалну симулацију са генералисаним континумом су представили Фејел и други у раду [8]. У раду коју су представили Гавамејн и Симон [7] само деформације су потребне да би неуронска мрежа предвидела напон, међутим представљени модели су једноставнији од биофизичких модела мишића. У току израде ове дисертације испоставило се да је за креирање сурогат модела мишића потребно више улазних променљивих, као и да су потребни додатни механизми учења, да би се сурогат модел понашао слично као оригинални модел унутар вишескалне симулације.

1.2 Предмет и допринос дисертације

У овој дисертацији је приказано креирање сурогат модела заснованих на различитим неуронским мрежама. Један од приступа је заснован на рекурентним и конволуционим неуронским мрежама за анализу временских серија, при чему се за обучавање користе прикупљени подаци. Подаци су, у току израде дисертације, прикупљени из рачунарске симулације под називом *Mexie*, који представља програм за вишескалну анализу методом коначних елемената са имплементираним Хакслијевим моделом мишића на микронивоу. У току анализе методом коначних елемената, материјални модел добија стреч, и даје напон и тренутну крутост на основу стања и параметара материјала. Дакле, стреч и мишићна активација у тренутном временском кораку су неопходни за рачунање напона и тренутне крутости, али нису довољни јер је понашање мишића условљено историјом оптерећења. Да би историја оптерећења била узета у обзир, креирана је временска серија која се састоји од активације у тренутном и претходним корацима, стреча у тренутном и претходним корацима, напона у претходним временским корацима и тренутне крутости у претходним временским корацима. На основу описане временске серије, рекурентне и конволуционе неуронске мреже предвиђају инкремент напона и тренутне крутости за тренутни временски корак. Колико је аутору дисертације познато, сурогат модел Хакслијевог модела мишића, такав да он ради у динамичком окружењу какве су вишескалне симулације методом коначних елемената, није направљен раније.

Други приказан приступ за креирање сурогат модела је заснован на неуронским мрежама подржаним физичким законима, где је вишеслојни перцептрон обучаван тако да апроксимира решење Хакслијеве парцијалне диференцијалне једначине за мишићну контракцију. На основу тренутног и претходног издужења, као и тренутне мишићне

активације, неуронска мрежа даје апроксимацију решења Хакслијеве једначине за контракцију, а затим се рачунају напони и изводи напона.

Mexie је, поред прикупљања података, коришћен и за тестирање понашања сурогат модела, односно поређење са оригиналним Хакслијевим моделом. *Mexie* пружа могућност за рад са дводимензионалним елементима, али није погодан за спровођење симулација понашања леве коморе, па су због тога, након тестирања, сурогат модели уграђени и у *PAK*, програм за анализу конструкција заснован на методи коначних елемената. Уградњом сурогат модела у *PAK*, специјално *PAK-FIS*, који поред модула за механику, поседује и модуле за транспорт и електрофизиологију, омогућена је анализа одзива леве коморе на различите побуде.

Допринос ове дисертације је најпре методолошки, јер су уведени поступци за креирање сурогат модела мишића, као и поступци за њихову интеграцију у софтверски оквир за анализу методом коначних елемената. Постигнути су резултати који показују да сурогат модел даје веома слична решења као оригинални модел, а притом је десетинама пута бржи од њега. На крају, сурогат модел је примењен за симулирање контракције леве коморе, што би било веома тешко, можда и немогуће, са оригиналним Хакслијевим моделом.

1.3 Основне хипотезе дисертације

У циљу развоја сурогат модела биофизичких модела мишића и њихове ефикасније употребе у анализи методом коначних елемената, уведене су следеће претпоставке:

- Неопходне и довољне улазне величине сурогат модела, које се добијају од макронивоа, а на основу којих ће бити прорачунат активни напон и тренутна крутост мишића у текућем временском кораку анализе методом коначних елемената (МКЕ) су:
 - ❖ Активација мишића у текућем временском кораку и у претходним временским корацима МКЕ анализе,
 - ❖ Издужење мишићног влакна у текућем временском кораку и у претходним временским корацима МКЕ анализе,
 - ❖ Напон и извод напона у претходним временским корацима МКЕ анализе.
- Биофизички модели мишића су уграђени у програме под називом *Musico* и *Mexie*. Претпоставка је да су ови модели довољно тачни.
- Из софтверских пакета наведених у претходној претпоставци се прикупљају вредности улазних и излазних величина наведених у првој претпоставци. На основу прикупљених вредности, креирају се сурогат модели обучавањем различитих типова вештачких неуронских мрежа.
- Вредности за активацију мишића и издужење мишићног влакна, морају бити доступне у програму за МКЕ анализу у свакој интеграционој тачки.
- Скуп параметара, који се односе на мутације протеина, ће бити константан у току креирања сурогат модела и у току МКЕ анализе. Ови параметри се не користе као улазне величине сурогат модела. Овај сет параметара је унапред одређен.
- Након исправно спроведене обуке, у симулацијама од интереса, сурогат модели дају резултате који не одступају за више од 5% од оних које дају оригинални биофизички модели.
- Сурогат модели су рачунски ефикаснији од оригиналних модела које замењују, троше мање рачунарског времена и меморије.

1.4 Преглед садржаја дисертације

Ова дисертација садржи следећих девет поглавља:

1. Увод
2. Феноменолошки и биофизички модели мишића
3. Нумеричке методе за моделирање механичког одзива мишића
4. Сурогат модели
5. Вештачке неуронске мреже
6. Креирање сурогат модела биофизичких модела мишића
7. Примена сурогат модела у анализи методом коначних елемената
8. Резултати и дискусија
9. Закључна разматрања

У првом поглављу је дефинисан предмет и циљ дисертације, дате су полазне хипотезе, као и преглед литературе са досадашњим истраживањима у области моделирања мишића и области креирања сурогат модела. У другом поглављу су приказани материјални модели мишића. У трећем поглављу су описане нумеричке методе које се користе приликом анализирања одзива мишића: методе за решавање једначина којима се описују конститутивне релације, метод коначних елемената итд. У четвртом поглављу је објашњен појам сурогат моделирања, заједно са техникама које се користе за креирање сурогат модела. У петом поглављу су описани различити типови вештачких неуронских мрежа и алгоритми који се користе приликом обучавања мрежа. У шестом поглављу је описан поступак за креирање сурогат модела биофизичких модела мишића. У седмом поглављу је описан процес интеграције сурогат модела у програме за анализу методом коначних елемената. У осмом поглављу су приказани резултати, дата су поређења оригиналних биофизичких модела мишића и сурогат модела, при чему се пореде добијени напони, изводи напона, брзина прорачуна и потрошња меморије. У деветом поглављу су дата закључна разматрања ове дисертације као и могући правци будућих истраживања.

2 Феноменолошки и биофизички модели мишића

У овом поглављу ће бити дата класификација мишићних контракција, биће описани типови мишићних ткива, филаменти од којих се састоје саркомере и генерисање силе на основу теорије клизећих филамената, а затим ће бити описани материјални модели мишића.

Мишићи су органи чија је основна функција да генеришу силу неопходну за кретање тела. Под утицајем нервне стимулације мишићи се контракују и савладавају спољашње оптерећење. Тетанизовано стање мишића је стање у ком мишићи генеришу максималну силу, која остаје константна у току времена. Максимални напон који се генерише у току контракције је вишеструко већи од напона мишића који није стимулисан односно није активиран. Мишићне контракције се могу поделити на трзаје и тетаничке контракције. Трзај је контракција, изазвана стимулацијом, у току које мишић не достиже максималну силу, већ крене да се опушта пре достигања максималне вредности. Тетаничка контракција је тип контракције у току које је стимулација довољно дуга, тако да мишић достиже максималну силу. Услед различитих утицаја мишић може да се скрати, издужи, или да остане исте дужине. У односу на промену дужине мишића, контракције се могу поделити на концентричне, ексцентричне и изометријске контракције. Концентрична контракција је контракција у току које се мишић скраћује. У овом случају, генерисана сила је јача од отпора. Пример концентричне контракције је подизање терета: бицепс произведи силу која је већа од тежине терета и скраћује се. Код ексцентричне контракције мишић се издужује, и отпор је већи од генерисане силе у мишићу. Пример ексцентричне контракције је спуштање терета: рука се помера надолу, терет полако спушта, а мишић се издужује. У току изометријске контракције мишић остаје непромењене дужине, генерисана сила је једнака спољашњој. Пример је држање терета без померања.

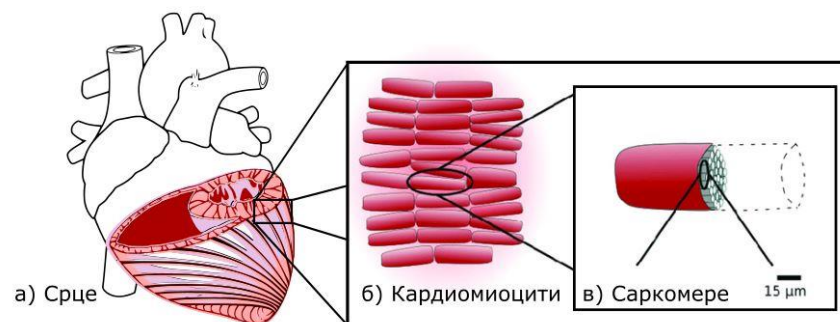
У телу постоје различити типови мишићног ткива: скелетни, глатки и срчани. Шематски приказ мишићних ћелија различитог типа је дат на слици 2.1.



Слика 2.1 Структура скелетних, глатких и срчаних мишићних ћелија

Скелетни мишићи су део мишићно-скелетног система. Скелетни мишићи су везани за кости тетивама, и заједно са костима пружају потпору телу и омогућавају му кретање. Ови мишићи спадају у групу вољних мишића, који су такође познати и као попречно-пругасти мишићи. Контракцију вољних мишића регулише вољна мождана активност. Попречно-пругасте мишиће чине танке дугачке ћелије цилиндричног облика, које се називају мишићним влакнима. Влакна су постављена паралелно и обложена су слојем растреситог везива, под називом ендомизијум. Већи број влакана се удружује и формира сноп, који је окружен омотачем под називом перимизијум. На крају, ови

снопови формирају мишић и окружени су још једним омотачем изграђеним од густог везивног ткива, који носи назив епимизијум. Глатки мишићи се налазе у органима као што су бешика, душник, материца, очи итд. Ови мишићи су инервисани аутономним нервним системом што значи да не раде под утицајем наше воље и свести. Глатки мишићи су способни за дуготрајне контракције и тешко се замарају. Глатке мишићне ћелије су вретенастог облика. Срчано мишићно ткиво чини средишњи слој срчаног зида, који се назива миокардом, и средишњи слој плућних вена у близини ушћа у срце. По структури срчани мишићи су слични попречно-пругастим, а по функцији слични су глатким мишићима. Срчано мишићно ткиво је састављено од попречно-пругастих влакана, док се срчана мишићна влакна састоје од серијски везаних ћелија, које се називају кардиомиоцитима. Унутар кардиомицита се налазе саркомере. На слици 2.2 је приказано срце, са кардиомиоцитима (Слика 2.2б), унутар којих се налази сноп саркомера (Слика 2.2в).

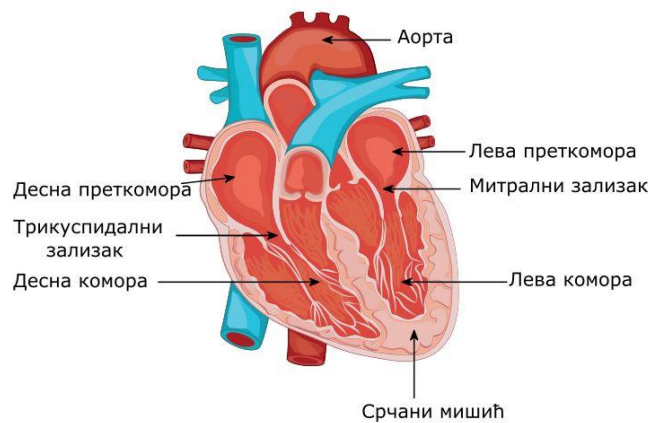


Слика 2.2 Шематски приказ структуре срчаног мишића од нивоа органа до нивоа саркомере

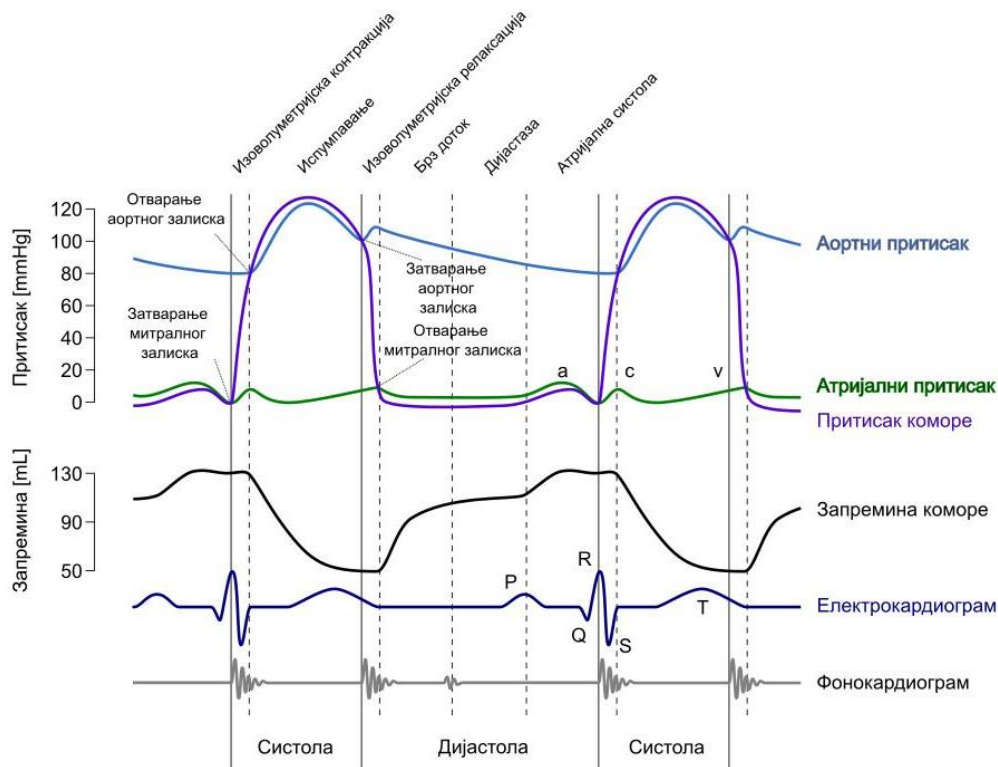
Срце се састоји од леве и десне коморе и леве и десне преткоморе. Шематски приказ срца са коморама и преткоморама дат је на слици 2.3 [9]. Лева комора и преткомора су раздвојене митралним залистком, док су десна комора и преткомора раздвојене трикуспидалним залистком. Срце се скупља и шири у току једног срчаног циклуса, који се понавља око 70 пута у минути. У току ширења, које се назива дијастолом, крв дотиче из леве преткоморе у леву комору, док је митрални залистак отворен. У исто време крв дотиче у десну комору из десне преткоморе, и трикуспидални залистак је отворен. У току дијастоле, аортни залистак, који се налази између аорте и леве коморе, и пулмонални залистак, који се налази између пулмоналне артерије и десне коморе, су затворени. У току систоле, коморе се контрахују услед мишићне активације, аортни и пулмонални залисци се отварају, услед чега долази до испумпавања крви. У току систоле митрални и трикуспидални залисци су затворени.

На слици 2.4 је дат Вигерсов дијаграм, који је увео Карл Вигерс (1883-1963) познат по истраживањима у области срца, посебно срчаног притиска [10]. Дијаграм укључује графике притисака, запремине, електрокардиограма и фонокардиограма. На крају дијастоле митрални залистак се затвара, аортни залистак остаје затворен, а мишићи услед активације генеришу силу и повећавају притисак унутар затворене коморе. Ово додатно оптерећење и повећавање притиска без промене запремине коморе се назива изоволуметријском контракцијом. На крају систоле аортни залистак се затвара,

митрални залистак остаје затворен, мишићни напон опада, што доводи до пада притиска без промене запремине. Овај кратак период се назива изоволуметријском релаксацијом. На Вигерсовом дијаграму се види да је притисак унутар леве коморе висок у току систоле, а низак у току дијастоле. Запремина расте у току дијастоле, а опада у току систоле. Фонокардиограм се добија услед вибрација срчаних залистака у току њиховог затварања. Електрокардиограм приказује пропацију електричног сигнала.



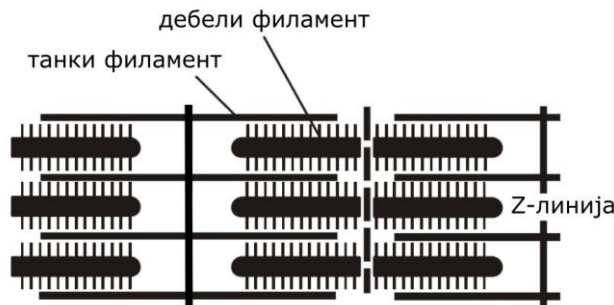
Слика 2.3 Шематски приказ срца са коморама и залистима [9]



Слика 2.4 Вигерсов дијаграм са притисцима, запремином, ехокардиограмом и фонокардиограмом у току срчаног циклуса [10]

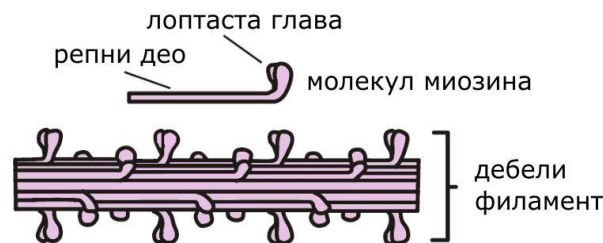
Саркомера представља основну контрактилну јединицу мишића. Она се састоји од танких (актин) и дебелих (миозин) влакана. Актин и миозин се већим делом састоје од

протеина по којима су добили називе. Саркомера је ограничена Z-линијама. Шематски приказ саркомере је дат на слици 2.5.



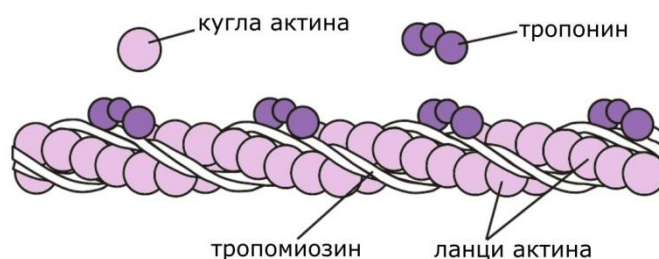
Слика 2.5 Шематски приказ структуре саркомере

У средини саркомере се налазе миозинска односно дебела влакна. Један молекул миозина са састоји од дугачког дела, направљеног од лаког мермоиозина, и од лоптастог дела, направљеног од тешког мермоиозина. Лоптасте главе се издвајају из влакна у паровима (Слика 2.6). На свакој глави миозина постоји простор за везивање за актинско место и простор за АТФ (аденозинтрифосфат) који ослобађа енергију, потребну за мишићну контракцију. Када се миозинска глава веже за актинско место, она се назива попречним мостом (енгл. corss-bridge), јер остварује везу између два филамента.



Слика 2.6 Шематски приказ структуре миозина

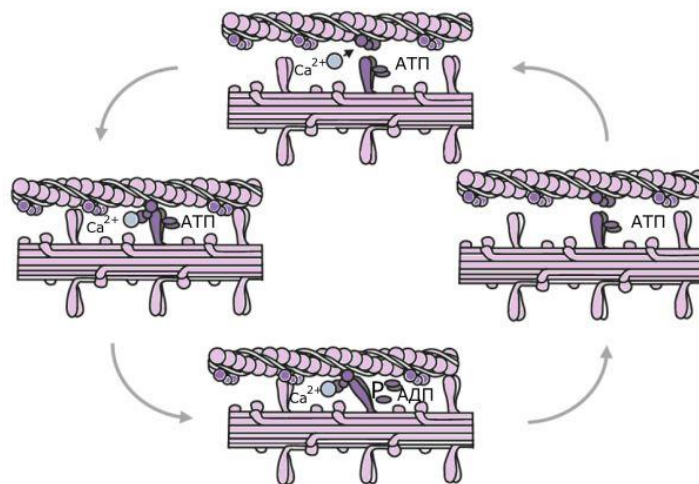
Главни део актинских влакана је сачињен од два умотана ланца актина. Поред актина, ова влакна садрже и молекуле тропомиозина и тропонина. Тропомиозин лежи у жлебу, који формирају ланци актина, а тропонин се налази дуж ланца у правилним размацама. Структура актина је приказана на слици 2.7.



Слика 2.7 Шематски приказ структуре актина

Теорија клизећих филамената, такође позната под називом теорија попречних мостова, је једна од најприхваћенијих теорија генерисања силе у мишићу на молекуларном

нивоу. Основе ове теорије је поставио А.Ф. Хаксли. Сматра се да је скраћење мишића последица клизања актинског филамента дуж миозинских. Ово померање филамената је последица интеракција миозинских глава са актином, које се одвијају у такозваним циклусима попречних мостова (енгл. cross-bridge cycle). Главе миозинског влакна се везују за актинска влакна на местима који се називају актин сајтови. Након качења, производња силе и покрети се јављају услед ротације главица. Миозинске главице вуку актинска влакна ка средини саркомере. У току стимулације мишића се ствара акциони потенцијал којим започиње ослобађање јона калцијума Ca^{2+} . Калцијум се везује за тропонин С који се налази на актинском филаменту. Услед везивања калцијума, тропомиозин се помера ван лежишта и открива актин сајтове на актинским нитима, за које се могу везати миозинске главице. Овим откривањем активних места на актину, настаје услов за започињање циклуса попречних мостова и за настајање контракције (Слика 2.8). За миозинске главе се вежу АТП молекули који се разграђују на неоргански фосфат и аденозиндифосфат (АДП), али остају везани уз главицу. При разградњи се мења положај главе, која се усправља ка актинском филаменту. Овако уздигнута глава може да се закачи за актински филамент у тренутку када су актински сајтови откривени. Када се веже за актински филамент, главица се нагиње ка репу и повлачи актинско влакно ка средини саркомере. Овај замах, изазива контракцију односно скраћивање, и приликом њега се отпуштају АДП и фосфат, чиме се ослобађа место на главици за везивање новог АТП молекула. Када се нов АТП молекул веже за главу, она се раздваја од актинског филамента. Процес качења, замаха и раздвајања се понавља док су актински сајтови откривени или док је померање актинског филамента низ миозински ланац могуће.



Слика 2.8 Циклус попречних мостова

2.1 Материјални модели мишића

За анализу механичког одзива мишића потребан је материјални модел, који даје силу односно напон, као одзив на деформацију. Материјални модели мишића се могу поделити на феноменолошке и биофизичке. Феноменолошки модели се ослањају на одређивање односа између улазних и излазних параметара модела емпиријским путем.

Код биофизичких модела, карактеристике и механички одзив мишића, се одређују на основу унутрашњих процеса, који се одвијају у микроструктури, најчешће на нивоу саркомере. Биофизички модели су прецизнији, али се ретко користе у пракси, јер су рачунски знатно захтевнији од феноменолошких. Типичан пример феноменолошког модела је Хилов модел, па се често феноменолошки модели називају моделима Хиловог типа. Хилова једначина представља везу између напона и брзине контракције:

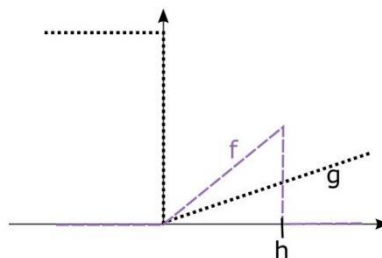
$$\frac{S}{S_0} = \frac{1 - \frac{v}{v_0}}{1 + c \frac{v}{v_0}} \quad (3.1)$$

где је S напон у мишићу, $c = S_0/a$, a , v_0 и S_0 су константе добијене емпиријским путем, v је брзина контракције, v_0 максимална брзина која се развија при напону $S = 0$ и S_0 је максимални изометријски напон. Без обзира на модификације, модели Хиловог типа користе неколицину макроскопски мерених параметара и не узимају у обзир промене напона мањих редова величине. Модели Хиловог типа су неосетљиви на fine промене, нарочито када је деформација неуниформна.

Типичан пример биофизичког модела је Хакслијев модел, па се за биофизичке моделе често каже да су модели Хакслијевог типа. Хаксли је разматрао динамику филамената унутар мишића и вероватноћу успостављања веза миозинских глава за актински филамент унутар саркомере [11, 12]. Функција $n(x, t)$ описује број успостављених веза између миозинских глава и актинског филамента (попречни мост), као функцију позиције најближег доступног актинског сајта за везивање релативно у односу на уравнотежену позицију миозинске главе x :

$$\frac{\partial n(x, t)}{\partial t} - v \frac{\partial n(x, t)}{\partial x} = [1 - n(x, t)]f(x, a) - n(x, t)g(x), \forall x \in \Omega \quad (3.2)$$

где $f(x, a)$ и $g(x)$ представљају стопе успостављања и раскидања попречних мостова респективно, v је брзина клизајућих филамената, позитивна у правцу контракције, а a је мишићна активација дата као функција времена [11]. Примери ових функција су дати на слици 2.9. Функција f расте када је x у интервалу од 0 до h , а за остале вредности x , f је нула. Издужење попречног моста је ограничено и зато након одређене вредности функција успостављања везе пада на нулу. Функција за раскидање везе g има високу вредност када је $x < 0$, а када је $x > 0$ функција g расте, што се односи на откачињање попречних мостова када су превише издужени. У пракси се може, при дефинисању функције закачињања, узимати у обзир и функција активације мишића (концентрација калцијума).



Слика 2.9 Шематски приказ стопе успостављања и раскидања попречних мостова

Парцијална диференцијална једначина (3.2) може бити решена методом карактеристика са почетним условом $n(x, 0) = 0$. Када су вредности $n(x, t)$ добијене, решавањем једначине (3.2), генерисана сила F унутар мишићних влакана као и специфична крутост K могу бити срачунате коришћењем формула:

$$F(t) = k \sum_{-\infty}^{\infty} n(x, t) x \, dx \quad (3.3)$$

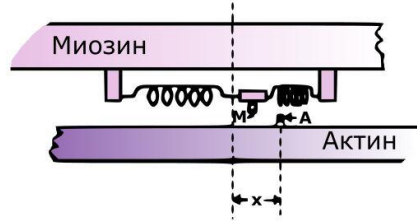
$$K(t) = k \sum_{-\infty}^{\infty} n(x, t) \, dx \quad (3.4)$$

где је k крутост попречних мостова. Напон и извод напона (тренутна крутост) се рачунају следећим формулама:

$$\sigma_m = F \frac{\sigma_{iso}}{F_{iso}} \quad (3.5)$$

$$\frac{\partial \sigma_m}{\partial e} = \lambda L_0 K \frac{\sigma_{iso}}{F_{iso}} \quad (3.6)$$

где F_{iso} представља максималну силу добијену у току изометријске контракције, σ_{iso} је максимални напон добијен у току изометријске контракције, L_0 почетна дужина саркомере, а λ је стреч. Шематски приказ Хакслијевог модела дат је на слици 2.10.



Слика 2.10 Шематски приказ Хакслијевог модела

Мишићи се активирају под утицајем калцијума. Мишићна активација може бити изведена из функције концентрације калцијума. За генерисање функције концентрације калцијума користи се параметарска једначина [13]:

$$Ca_i(t) = Ca_0 + (Ca_{max} - Ca_0) \frac{t}{\tau Ca} e^{1 - \frac{t}{\tau Ca}} \quad (3.7)$$

где $Ca_i(t)$ представља временски зависну међућелијску концентрацију Ca^{2+} , која има минималну вредност у Ca_0 и максималну вредност у Ca_{max} у тренутку $t = \tau Ca$. Концентрација калцијума може бити конвертована у мишићну активацију α коришћењем формуле:

$$\alpha = \frac{(Ca)^n}{(Ca)^n + (C_{50})^n} \quad (3.8)$$

где је C_{50} вредност потребна да се достигне 50% максималне доступности везивог калцијума, срачуната коришћењем $pC_{50} = pC_{50ref}(1 + \beta_2(\lambda - 1))$; $C_{50} = 10^{6-pC_{50}}$ [μM], где је n дефинисано као:

$$n = n_{ref}(1 + \beta_1(\lambda - 1)) \quad (3.9)$$

где је λ стреч, а pC_{50ref} , n_{ref} , β_1 и β_2 су константе. Коришћене су вредности $n_{ref} = 5.2$, $pC_{50ref} = 6.18$, $\beta_1 = 1.95$ и $\beta_2 = 0.31$, које су преузете из рада [13]. У вишескалним симулацијама може бити потребно решавати Хакслијеву једначину (3.2) хиљадама пута, што може трајати неколико сати или дана.

3 Нумеричке методе за моделирање механичког одзива мишића

У овом поглављу ће најпре бити описан метод карактеристика који се користи за решавање Хакслијеве парцијалне диференцијалне једначине за мишићну контракцију. На крају ће бити дат и метод коначних елемената, као универзалнији метод за решавање парцијалних диференцијалних једначина, који се у овом раду користи за симулирање механичког одзива мишића на микронивоу.

3.1 Метод карактеристика

Метод карактеристика је техника за решавање хиперболичких парцијалних диференцијалних једначина [14]. Метода се обично примењује на једначине првог реда, а теоријски се може користити за било који тип хиперболичких парцијалних диференцијалних једначина. Ова метода укључује одређивање специјалних кривих, под називом карактеристичне криве, дуж којих парцијална диференцијална једначина постаје фамилија обичних диференцијалних једначина [14]. Ове једначине се решавају дуж карактеристичне криве, да би се добила решења парцијалне диференцијалне једначине. Метод карактеристика може бити примењен на линеарне, семилеарне или квазилинеарне парцијалне диференцијалне једначине. Нека је дата квазилинеарна парцијална диференцијална једначина првог реда у генералној форми:

$$a(x, y, u) \frac{\partial u}{\partial x} + b(x, y, u) \frac{\partial u}{\partial y} = c(x, y, u) \quad (3.1)$$

у домену D , за променљиву u . Нека су први изводи a, b, c континуални по x, y, u . Графички, могуће решење једначине (3.1) у форми $u = u(x, y)$, представља површ S у \mathbf{R}^3 , Површ S може бити имплицитно представљена у форми једначине $f(x, y, u) = 0$, а решење једначине (3.1) се може записати као $f(x, y, u) = u(x, y) - u$. Нормала на површ $f(x, y, u) = 0$ је дата са ∇f . Дакле, једначина вектора нормалног на површ S је дата са:

$$\nabla f = \frac{\partial f}{\partial x} \hat{i} + \frac{\partial f}{\partial y} \hat{j} + \frac{\partial f}{\partial u} \hat{k} = \frac{\partial u(x, y)}{\partial x} \hat{i} + \frac{\partial u(x, y)}{\partial y} \hat{j} - \frac{\partial u}{\partial u} \hat{k} \equiv \left(\frac{\partial u}{\partial x}, \frac{\partial u}{\partial y}, -1 \right) = (u_x, u_y, -1) \quad (3.2)$$

Вектор $(u_x, u_y, -1)$ представља нормални вектор на било коју тачку површи $u = u(x, y)$. Нека је $\bar{A} = a\hat{i} + b\hat{j} + c\hat{k}$ векторско поље, где су a, b, c из једначине (3.10). Парцијална диференцијална једначина (3.10) може бити записана као $\bar{A}\nabla f = 0$ односно:

$$(a\hat{i} + b\hat{j} + c\hat{k})\left(\frac{\partial u}{\partial x}\hat{i} + \frac{\partial u}{\partial y}\hat{j} - \hat{k}\right) = 0 \quad \text{или} \quad (a, b, c)(u_x, u_y, -1) = 0 \quad (3.3)$$

Једначина (3.3) показује да су вектори (a, b, c) и $(u_x, u_y, -1)$ ортогонални, и пошто је вектор $(u_x, u_y, -1)$ нормалан на површ $u = u(x, y)$, вектор $\bar{A} = (a, b, c)$ мора бити тангентни вектор на површ S у свакој тачки (x, y, u) . Дакле, \bar{A} дефинише векторско поље у простору (x, y, u) , које је тангентно на график решења једначине (1) у свакој тачки (x, y, u) . Површи које су тангентне на векторско поље у свакој тачки у \mathbf{R}^2 се називају интегралним површима векторског поља. Дакле, површ $f(x, y, u) = u(x, y) - u = 0$ у (x, y, u) простору представља решење једначине (3.1) ако и само ако правац

векторског поља $\bar{A} = (a, b, c)$ лежи у тангентној равни интегралне површи $f(x, y, u) = 0$ у свакој тачки (x, y, u) . Вектор (a, b, c) одређује правац, који се назива карактеристичним правцем. Пошто је вектор \bar{A} тангента на површ у свакој тачки (x, y, u) важи:

$$\bar{A} \times ds = \begin{vmatrix} \hat{i} & \hat{j} & \hat{k} \\ a & b & c \\ dx & dy & du \end{vmatrix} = 0 \quad (3.4)$$

где је $ds = dx\hat{i} + dy\hat{j} + du\hat{k}$ пројекција вектора \bar{A} на порш S . Развијањем детерминанте добија се једначина:

$$(bdu - cdy)\hat{i} + (cdx - adu)\hat{j} + (ady - bdx)\hat{k} = 0 \quad (3.5)$$

која доводи до такозваних *Lagrange-Charpit* једнакости:

$$\frac{dx}{a} = \frac{dy}{b} = \frac{du}{c} \quad (3.6)$$

Дакле, решење квазилинеарне једначине се може изразити описом тангентне равни, која се описује нагибом:

$$\frac{du}{dx} = \frac{c}{a} \quad \text{и} \quad \frac{du}{dy} = \frac{c}{b} \quad (3.7)$$

или еквивалентно, једном једначином из претходног пара са:

$$\frac{dx}{dy} = \frac{a(x, y, u)}{b(x, y, u)} \quad (3.8)$$

Једначина (3.8) дефинише фамилију кривих у (x, y, u) простору које припадају површи која је решење једначине (1). Ове криве се зову карактеристике и сет једначина (3.7, 3.8) се назива карактеристичним једначинама квазилинеарне парцијалне диференцијалне једначине (3.1). Пројекција карактеристичне криве на (x, y) -раван (када је $u = 0$) се назива карактеристична базна крива, пројектована карактеристична крива или карактеристика. Може се приметити да су само две од три диференцијалне једначине из система (3.7, 3.8) независне. Решавањем ових једначина добија се фамилија карактеристичних кривих у (x, y, u) . Кроз сваку тачку (x, y, u) у простору постоји јединствена карактеристична крива и тангентни вектор (a, b, c) . Нека су $x = x(s)$, $y = y(s)$, $u = u(s)$ параметарске једначине, дате у зависности од параметра s . Увођењем параметра s , *Lagrange-Charpit* једначине постају:

$$\frac{dx}{a} = \frac{dy}{b} = \frac{du}{c} = ds \quad (3.9)$$

и тада систем обичних диференцијалних једначина може бити записан као:

$$\frac{dx}{ds} = a, \quad \frac{dy}{ds} = b, \quad \frac{du}{ds} = c \quad (3.10)$$

где је s мера растојања дуж криве. Систем обичних парцијалних диференцијалних једначина (3.10) чини карактеристичне једначине парцијалне диференцијалне једначине (3.1) у параметарској форми. Фамилија кривих $x = x(s)$, $y = y(s)$, $u = u(s)$ одређена решавањем система једначина (3.19) се назива карактеристичним кривама ПДЈ (1). Под претпоставкама глаткоће a, b, c , постоји тачно једна површ $(x(s), y(s), u(s))$ која је решење (3.9) и која пролази кроз дату тачку (x_0, y_0, u_0) у простору (x, y, u) . Често није од интереса опште решење парцијалне диференцијалне једначине већ специфично решење $u = u(x, y)$ дуж криве Γ која пролази кроз дату тачку (x, y) , обично (x_0, y_0) када је $s = 0$. Нека је дата општа квазилинеарна диференцијална једначина:

$$\sum_{i=1}^n a_i(x_1, x_2, \dots, x_n, u) \frac{\partial u}{\partial x_i} = c(x_1, x_2, \dots, x_n, u) \quad (3.11)$$

Параметарска форма карактеристичних кривих једначине (3.11) се добија решавањем система обичних диференцијалних једначина:

$$\begin{aligned} \frac{dx_i}{ds} &= a_i(x_1, x_2, \dots, x_n, u) \\ \frac{du}{ds} &= c(x_1, x_2, \dots, x_n, u) \end{aligned} \quad (3.12)$$

где је s параметарска променљива. Једначине (3.12) су карактеристичне једначине ПДЈ (3.11).

3.1.1 Итеративни поступак

Нека су дате једначине $adt - bdx = 0$ и $cdx - adu = 0$, односно $\frac{dx}{a} = \frac{dt}{b} = \frac{du}{c}$. Некад се ове једначине не могу решити аналитичким путем, па се за њихово решавање користе итеративне процедуре. Нека је вредност u позната у N тачака $T_m^{(0)}$ ($m = 1, 2, \dots, N$) у (x, t) -равни. За сваку тачку $T_m^{(0)}$ се може одредити карактеристика C_m којој тачка $T_m^{(0)}$ припада. Поступак интеграције се спроводи дуж карактеристика C_m . Карактеристика C_m се апроксимира дужима $T_m^{(k)} T_m^{(k+1)}$ ($k = 1, 2, \dots$), где тачке $T_m^{(k)}$ припадају C_m . Нека $x_m^{(k)}, t_m^{(k)}, u_m^{(k)}, a_m^{(k)}, b_m^{(k)}, c_m^{(k)}$ представљају редом вредности x, t, u, a, b, c у $T_m^{(k)}$ и нека су те вредности познате. Ако претпоставимо да је познато $t_m^{(k+1)}$, тада се вредности $x_m^{(k+1)}$ и $u_m^{(k+1)}$ могу приближно срачунати следећим итеративним поступком:

$$1. x_m^{(k+1)} = x_m^{(k)} + \frac{a_m^{(k)}}{b_m^{(k)}} (t_m^{(k+1)} - t_m^{(k)}) \quad (3.13)$$

$$2. u_m^{(k+1)} = u_m^{(k)} + \frac{c_m^{(k)}}{a_m^{(k)}} (x_m^{(k+1)} - x_m^{(k)}) \quad (3.14)$$

3. На основу $x_m^{(k+1)}, u_m^{(k+1)}, t_m^{(k+1)}$ се одређују $a_m^{(k+1)}, b_m^{(k+1)}, c_m^{(k+1)}$.

4. Коригована вредност $x_m^{(k+1)}$ се добија из

$$\frac{1}{2} (a_m^{(k+1)} + a_m^{(k)}) (t_m^{(k+1)} - t_m^{(k)}) - \frac{1}{2} (b_m^{(k+1)} + b_m^{(k)}) (x_m^{(k+1)} - x_m^{(k)}) = 0 \quad (3.15)$$

Коригована вредност $u_m^{(k+1)}$ се добија из

$$\frac{1}{2}(c_m^{(k+1)} + c_m^{(k)})(x_m^{(k+1)} - x_m^{(k)}) - \frac{1}{2}(a_m^{(k+1)} + a_m^{(k)})(u_m^{(k+1)} - u_m^{(k)}) = 0 \quad (3.16)$$

Кораци 3. и 4. се понављају док корекције не буду мање од задате прецизности.

3.2 Метод коначних елемената

Метод коначних елемената (МКЕ) је техника за приближно решавање парцијалних диференцијалних једначина са задатим почетним и граничним условима. Пошто се парцијалним диференцијалним једначинама описују најразличитији физички системи, метод коначних елемената је веома популаран у области инжењерске анализе и дизајна. Први радови из области методе коначних елемената појавили су се четрдесетих година двадесетог века. Утемељитељи методе су Клаф, Мартин, Топ и Тарнер који су направили основни концепт методе коначних елемената. Сва позната сазнања у подручју анализе коначним елементима су 1960. сумирана на конференцији *US Air Force*, где је договорено да се направи и први софтвер *Nasa Structural Analysis*. На Универзитету у Берклију развијен је *Structural Analysis Program (SAP)* од стране сарадника професора Клафа. Први универзитетски уџбеник написао је Кук 1974. године, у време када је метод већ био прихваћен. Посебан значај у развоју МКЕ имали су варијациони принципи механике континуума који су примењени на формулацију МКЕ, па је метод добио општи приступ. Даљи развој одвијао се у правцу раванских елемената. Тако је Ричард Курант решавајући граничне проблеме увијања предложио и користио троугаоне елементе, а решење добио помоћу Рицове варијационе методе. Хелингер и Рејзнер постављају мешовити модел коначних елемената у коме се комбиновано јављају силе и деформације као непознате величине.

Основна идеја анализе методом коначних елемената јесте дискретизација домена на поддомене на које се примењују знања из области механике континуума и нумеричке анализе. Поддомени, којима се дискретизује домен од интереса, се називају коначним елементима. Дискретизација домена на већи број коначних елемената била је лимитирајући фактор све до појаве аутоматских генератора мреже коначних елемената. Мрежа коначних елемената покрива цео домен који се моделира и чине је једноставни геометријски облици као што су троуглови и четвороуглови у дводимензионалном простору, или различити полиедри у тродимензионалном простору. Захваљујући декомпозицији на мање целине, метод се може применити на домене сложене геометрије и структуре. Код методе коначних елемената неопходно је да се оствари континуитет између елемената у мрежи, који је постигнут увођењем интерполационих функција, развијених захваљујући развоју математичке теорије сплајнова. Елементи су повезани чворовима, а њихов међусобни однос је дефинисан конститутивним једначинама материјала, законима о одржању енергије, масе и количине кретања. Успостављањем равнотежних једначина за коначне елементе и последично целокупну структуру, долази се до система једначина, који се може решити директним или итеративним методама. Анализа методом коначних елемената подразумева нумеричку симулацију одзива континуума на задате побуде. Физичке величине које су обухваћене моделом добијају се у дискретном облику, односно у чворовима коначних елемената. Са данашњим могућностима рачунара, након завршене анализе методом коначних

елемената, могуће је визуализовати поља напона, деформација, термичко поље и слично у зависности од физичког проблема који се решава.

3.2.1 Нелинеарна анализа у механици солида

Мишићи имају сложену структуру, деформишу се под утицајем спољашњег оптерећења и под утицајем унутрашње побуде [15]. Пошто мишићи показују анизотропне карактеристике и могу трпети велике деформације, за анализу њиховог механичког одзива методом коначних елемената се користе методе нелинеарне анализе [16,17]. Налажење равнотежног стања је основни проблем приликом симуирања механичког одзива материјала који има нелинеарне карактеристике. Услов равнотеже система коначних елемената се може изразити као:

$${}^t \mathbf{F}^{ext} - {}^t \mathbf{F}^{int} = 0 \quad (3.15)$$

где је ${}^t \mathbf{F}^{ext}$ вектор спољашњих чворних сила, а ${}^t \mathbf{F}^{int}$ вектор чворних сила које одговарају напонима елемената у конфигурацији у тренутку t . Даље, важи:

$${}^t \mathbf{F}^{ext} = {}^t \mathbf{F}_C^{ext} + {}^t \mathbf{F}_S^{ext} + {}^t \mathbf{F}_V^{ext} \quad (3.16)$$

где је ${}^t \mathbf{F}_C^{ext}$ вектор чворних, ${}^t \mathbf{F}_S^{ext}$ површинских, а ${}^t \mathbf{F}_V^{ext}$ вектор запреминских сила. Када се геометрија тела и карактеристике материјала нелинеарно мењају у току времена и зависе од историје оптерећења, често се користи такозвана инкрементална шема корак-по-корак. Најпре се временски интервал, у коме се систем посматра, дели на одговарајући број подинтервала. Под претпоставком да је решење у дискретном тренутку са почетка интервала познато, налази се решење за конфигурацију која одговара наредном дискретном временском тренутку. Ради једноставности можемо сматрати да су сви подинтервали исте дужине Δt , где је Δt погодно одабран временски корак. Ако је решење за конфигурацију у временском тренутку t познато, за конфигурацију у наредном дискретном тренутку $t + \Delta t$ једначина (3.15) гласи:

$${}^{t+\Delta t} \mathbf{F}^{ext} - {}^{t+\Delta t} \mathbf{F}^{int} = 0 \quad (3.17)$$

при чему за ${}^{t+\Delta t} \mathbf{F}^{int}$ важи:

$${}^{t+\Delta t} \mathbf{F}^{int} = {}^t \mathbf{F}^{int} + \mathbf{F}^{int} \quad (3.18)$$

где је \mathbf{F}^{int} прираштај сила у чворовима елемената који одговара прираштају померања и напона од тренутка t до тренутка $t + \Delta t$. Вектор \mathbf{F}^{int} се апроксимира употребом тангентне матрице, ${}^t \mathbf{K}$, која одговара материјалним и геометријским условима конфигурације у тренутку t :

$$\mathbf{F}^{int} \approx {}^t \mathbf{K} \mathbf{U} \quad (3.19)$$

где је \mathbf{U} вектор прираштаја померања у чворовима, и важи:

$${}^t \mathbf{K} = \frac{\partial {}^t \mathbf{F}^{int}}{\partial {}^t \mathbf{U}} \quad (3.20)$$

На основу (3.17), (3.19) и (3.20) добија се једначина:

$${}^t\mathbf{K}\mathbf{U} = {}^{t+\Delta t}\mathbf{F}^{ext} - {}^t\mathbf{F}^{int} \quad (3.21)$$

чијим се решавањем по \mathbf{U} добијају приближна померања у тренутку $t + \Delta t$, ${}^{t+\Delta t}\mathbf{U} = {}^t\mathbf{U} + \Delta\mathbf{U}$. На основу померања се рачунају напони и одговарајуће чворне силе, чије ће вредности, такође, бити приближно тачне. Због апроксимације се у пракси користе итеративни поступци којима се тражене вредности доводе до задовољавајуће тачности. Један од често коришћених итеративних метода јесте *Newton-Raphson*. Итеративна процедура према *Newton-Raphson* методи је дефинисана следећим једначинама:

$$\begin{aligned} {}^{t+\Delta t}\mathbf{K}^{(i-1)}\Delta\mathbf{U}^{(i)} &= {}^{t+\Delta t}\mathbf{F}^{ext} - {}^{t+\Delta t}\mathbf{F}^{int(i-1)} \\ {}^{t+\Delta t}\mathbf{U}^{(i)} &= {}^{t+\Delta t}\mathbf{U}^{(i-1)} + \Delta\mathbf{U}^{(i)} \end{aligned} \quad (3.22)$$

где је i број итерације, а почетни услови су ${}^{t+\Delta t}\mathbf{U}^{(0)} = {}^t\mathbf{U}$, ${}^{t+\Delta t}\mathbf{K}^{(0)} = {}^t\mathbf{K}$ и ${}^{t+\Delta t}\mathbf{F}^{int(0)} = {}^t\mathbf{F}^{int}$. Вектор ${}^{t+\Delta t}\mathbf{F}^{ext} - {}^{t+\Delta t}\mathbf{F}^{int(i-1)}$ се односи на силе које нису уравнотежене са напонима у елементима. Да би се ова неуравнотеженост елиминисала, потребно је одредити инкремент чворних померања. Корекција померања се врши у свакој итерацији. Поступак се зауставља када прираштаји неуравнотежених сила и прираштају померања буду довољно мали. Вектор унутрашњих сила ${}^{t+\Delta t}\mathbf{F}^{int(i-1)}$ и матрица крутости ${}^{t+\Delta t}\mathbf{K}^{(i-1)}$ се добијају на основу једначина:

$$\begin{aligned} {}^{t+\Delta t}\mathbf{F}^{int(i-1)} &= \int_{t+\Delta t V^{(i-1)}} ({}^{t+\Delta t}\mathbf{B}^T {}^{t+\Delta t}\boldsymbol{\sigma}^{(i-1)}) dV \\ {}^{t+\Delta t}\mathbf{K}^{(i-1)} &= \int_{t+\Delta t V^{(i-1)}} ({}^{t+\Delta t}\mathbf{B}^T {}^{t+\Delta t}\mathbf{C} {}^{t+\Delta t}\mathbf{B})^{(i-1)} dV \end{aligned} \quad (3.23)$$

где је ${}^{t+\Delta t}\boldsymbol{\sigma}^{(i-1)}$ напон, ${}^{t+\Delta t}\mathbf{C}^{(i-1)}$ тангентна конститутивна матрица, а ${}^{t+\Delta t}\mathbf{B}$ матрица извода интерполационих функција. На основу једначина (3.34) се може видети да је основни задатак одређивање напона и тангентне конститутивне матрице у одговарајућим тачкама материјала. У листингу испод је приказана итеративна шема.

Newton-Raphson итеративна шема

1. Иницијализација за тренутни временски корак

$$i = 0$$

$${}^{t+\Delta t}\mathbf{F}^{(0)} = {}^t\mathbf{F}, {}^{t+\Delta t}\mathbf{K}^{(0)} = {}^t\mathbf{K}, {}^{t+\Delta t}\mathbf{U}^{(0)} = {}^t\mathbf{U}$$

2. Итерација i

$$i = i + 1$$

$${}^{t+\Delta t}\mathbf{K}^{(i-1)}\Delta\mathbf{U}^{(i)} = {}^{t+\Delta t}\mathbf{F}^{ext} - {}^{t+\Delta t}\mathbf{F}^{int(i-1)}$$

$${}^{t+\Delta t}\mathbf{U}^{(i)} = {}^{t+\Delta t}\mathbf{U}^{(i-1)} + \Delta\mathbf{U}^{(i)}$$

3. Провера конвергенције

Критеријум силе

$$\| {}^{t+\Delta t}\mathbf{F}^{ext} - {}^{t+\Delta t}\mathbf{F}^{int(i)} \| \leq \epsilon_F \| {}^{t+\Delta t}\mathbf{F}^{ext} - {}^t\mathbf{F}^{int} \|$$

Критеријум померања

$$\| \Delta\mathbf{U}^{(i)} \| \leq \epsilon_D \| {}^{t+\Delta t}\mathbf{U}^{(i)} \|$$

Критеријум енергије

$$\Delta\mathbf{U}^{(i)T} ({}^{t+\Delta t}\mathbf{F}^{ext} - {}^{t+\Delta t}\mathbf{F}^{int(i)}) \leq \epsilon_E \Delta\mathbf{U}^{(i)T} ({}^{t+\Delta t}\mathbf{F}^{ext} - {}^t\mathbf{F}^{int})$$

Уколико је услов конвергенције задовољен прелази се на наредни временски корак, а у супротном, процедура се понавља од другог корака.

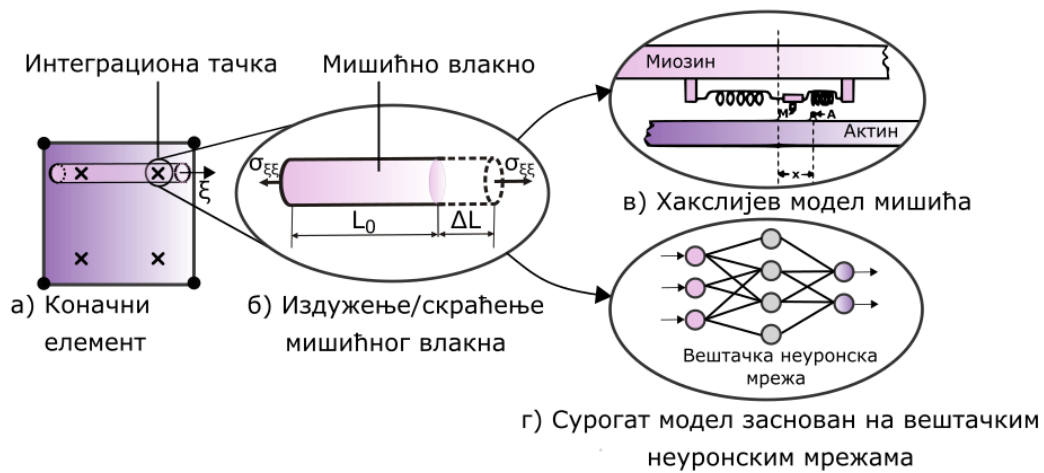
У инкрементално-итеративној шеми, уравнотежена конфигурација мишића може бити срачуната, узимајући у обзир мишиће као структуру састављену од влакана, која може да се контрахује под утицајем активације унутар деформабилног конективног ткива. Водеће једначине равнотеже структуре у деформисаној конфигурацији у тренутку (t) и итерацији (i) су:

$$({}^{t+\Delta t}K_{pass} + {}^{t+\Delta t}K_{act})^{(i-1)}\Delta U^{(i)} = {}^{t+\Delta t}F_{ext} + {}^{t+\Delta t}F_{pass}^{(i-1)} + {}^{t+\Delta t}F_{act}^{(i-1)} \quad (3.24)$$

где ${}^{t+\Delta t}F_{ext}$, ${}^{t+\Delta t}F_{pass}^{(i-1)}$, ${}^{t+\Delta t}F_{act}^{(i-1)}$ представљају векторе екстерних оптерећења, пасивних унутрашњих сила и активних молекуларних сила упакованих у силе у чворовима коначних елемената, респективно; ${}^{t+\Delta t}K_{pass}^{(i-1)}$ је матрица крутости пасивних компоненти и ${}^{t+\Delta t}K_{act}^{(i-1)}$ је крутост активних компоненти, која је у случају Хакслијевог модела кумулативна крутост веза актина и миозина; $\Delta U^{(i)}$ су инкрементална померања у чворовима у итерацији (i). Активне силе ${}^{t+\Delta t}F_{act}^{(i-1)}$ и крутост ${}^{t+\Delta t}K_{act}^{(i-1)}$ су директно зависни од брзине мишићне деформације у правцу мишићних влакана. Укупан напон $\bar{\sigma}$ се изражава преко активних мишићних сила и пасивних сила у конективним и неконтрактилним деловима ткива:

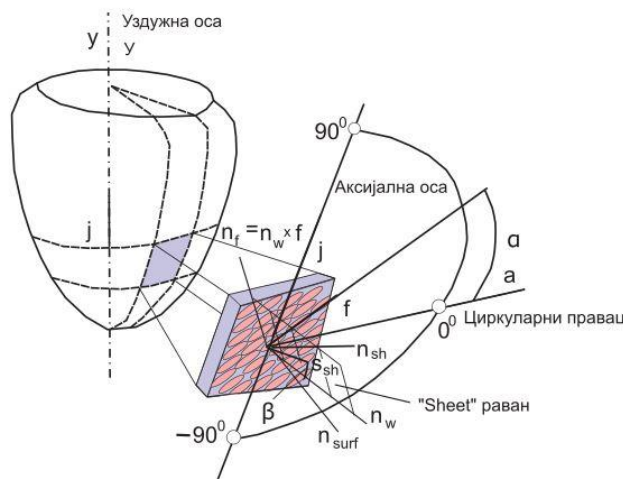
$$\bar{\sigma} = \phi \bar{\sigma}_m + (1 - \phi) \bar{\sigma}^E \quad (3.25)$$

где је ϕ удео мишићних влакана у укупном мишићној запремини, $\bar{\sigma}_m$ је активни напон генерисан у мишићима и $\bar{\sigma}^E$ је напон у пасивним деловима мишића. На слици 2.11 је приказан један коначни елемент са четири чвора и влакном у правцу ξ осе. Унутар интеграционе, или Гаусове, тачке посматрамо издужење односно скраћење влакна, на основу ког је потребно срачунати активни напон. За формирање тангентне конститутивне матрице неопходно је израчунати извод напона по деформацији, односно тренутну крутост мишића [18,19]. Зависност активног напона од издужења, представља конститутивну релацију мишића, која се може изразити неким материјалним моделом мишића, на пример Хилловим или Хакслијевим моделом [18, 19]. Материјални модел, унутар сваке интеграционе тачке домена од интереса, на основу стања материјала из претходног корака, тренутног стреча и материјалних параметара, даје напон и извод напона. На слици 2.11в је приказан Хакслијев модел, уместо ког се за одређивање активног напона и тренутне крутости, може користити сурогат модел [20]. На слици 2.11г сурогат модел представља рачунски ефикасну замену за оригинални Хакслијев модел и приказан је као вештачка неуронска мрежа. О креирању замене за оригинални Хакслијев модел, и његовој примени у анализи методом коначних елемената ће бити речи у наредним главама. За рачунање пасивног дела напона се често користи линеарно еластични модел. Код модела срца, за рачунање напона у пасивном делу ткива се може користити и експериментални Холзапфелов модел [21-24].



Слика 3.1 Коначни елемент са в) оригиналним Хакслијевим моделом и г) сурогат моделом на микронивоу

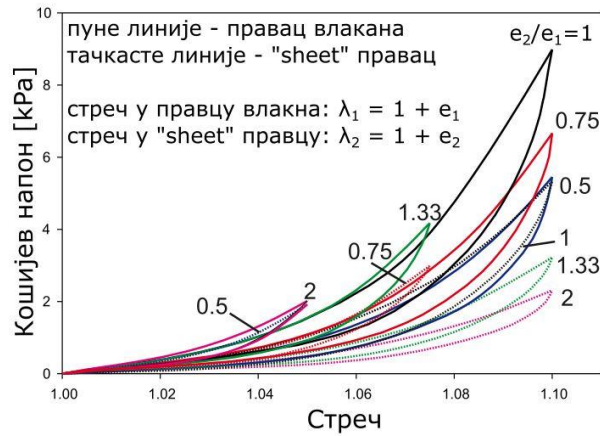
Срчани проблеми, болести и поремећаји су често повезани са ткивом леве коморе. Крв из леве коморе иде у артеријски систем, а из десне коморе у плућа. Пошто је отпор јачи приликом протока крви у артеријски систем, укупно оптерећење леве коморе је веће него оптерећење десне. Услед ових околности, проучавање процеса унутар леве коморе је од примарног интереса. Морфологија ткива леве коморе је веома комплексна. На слици 2.12 је дат шематски приказ леве коморе. Зид се састоји од слојева паралелних мишићних ћелија, а материјал зида леве коморе се може сматрати ортотропним, са векторима f и s у правцу влакна и $sheet$ правцу, респективно, који леже у равни тангентној на $sheet$ површ. Трећи правац n је дефинисан нормалом на $sheet$ раван, као што је приказано на слици 2.12.



Слика 3.2 Дефиниција вектора на зиду срчаног зида [25]

Конститутивне криве приказане на слици 2.13, су добијене експериментима са биаксијалним оптерећењем у правцу влакана (*mean fiber direction* - MFD) и *sheet* правцу (*mean sheet direction* - CFD). Ткиво је растезано до три нивоа стреча 1.05, 1.075 и 1.1 са одржавањем константних односа деформација e_2/e_1 , при чему је e_2 деформација у *sheet* правцу, а e_1 у правцу влакна. На слици 2.13, се види да су конститутивне криве нелинеарне, са хипереластичним карактеристикама, које су честе код биолошких

материјала. Однос напон-стреч зависи од нивоа стреча до ког је ткиво растезано и од односа деформација e_2/e_1 . Поред оптерећења, на слици 2.13 су приказане и криве напон-стреч за случајеве растерећења, када се ткиво враћа у почетни положај. Детаљи рачунске процедуре за рачунање напона, на основу датих експерименталних кривих, се могу видети у радовима [21-24]. Примери анализе одзива мишића се могу видети у радовима [26-43].



Слика 3.3 Конститутивне напон-стреч криве за ткиво леве коморе [25]

4 Суругат моделирање

Суругат моделирање је процес креирања модела који имитира понашање оригиналног система, али је рачунски ефикаснији од њега, у смислу да троши мање рачунарског времена и меморије. Суругат модел, метамодел или емулатор који добро апроксимира оригинални модел, може бити коришћен као његова замена. Многи инжењерски проблеми захтевају експерименте и/или рачунарске симулације које могу трајати неколико минута, сати или дана. С обзиром да је често потребно и неколико стотина или хиљада симулација да би се дизајн оптимизовао или анализирала нека физичка појава, убрзање извршавања симулација може бити од великог значаја. Један од могућих начина за убрзавање ових процеса, јесте коришћење суругат модела уместо симулације. Рачунарске симулације су дизајниране тако да предвиђају понашање физичког система решавањем одговарајућих једначина. Пошто је евалуација суругат модела знатно бржа него евалуација симулације, ови модели могу знатно убрзати анализу физичких процеса, по цену губитка прецизности, а некад и поузданости резултата. За суругат моделирање се често користе разне методе машинског учења као што су на пример: Кригинг методе, вештачке неуронске мреже, метод потпорних вектора, Фуријеово суругат моделирање, алгоритам насумичне шуме, Бајесове мреже итд.

Машинско учење је подобласт вештачке интелигенције, која за циљ има конструисање алгоритама и рачунарских система који су способни да се прилагођавају на аналогне нове ситуације и уче на основу искуства. По начину на који уче, алгоритми машинског учења се могу поделити на: (1) надгледане, (2) ненадгледане, (3) полу-надгледане и (4) алгоритме који уче подстицањем (добро понашање се награђује, или се лоше кажњава). Многи проблеми учења захтевају прикупљање података, који се користе за обучавање и за тестирање алгорита. Подаци се могу прикупљати из различитих извора, као што су базе података, екстерне агенције, рачунарске симулације итд.

Надгледано учење се односи на тип учења где постоји скуп података за обуку (улазни подаци) коју су обележени односно за које се зна излазна вредност (излазни подаци). На пример, улазни подаци за обуку могу бити слике, а ознаке могу бити називи животиња које се налазе на тим сликама. У току обуке алгорита, алгоритам апроксимира стварну функцију излазних података у зависности од улазних. Након обуке алгоритам даје модел који може да се користи за предвиђање излазних вредности и за улазе који нису коришћени у току обуке. Типични примери надгледаног учења су регресија и класификација. Код регресије, излазне вредности су реалне континуалне вредности, док су код класификације излазне вредности категорије (нпр. „здрав“ или „није здрав“). Регресиона анализа је скуп процедура помоћу којих се оцењује међусобна повезаност зависне променљиве Y и независних променљивих X_1, X_2, \dots, X_n где је n број независних променљивих. Независне променљиве се често називају и предикторским променљивама, регресорима или улазним атрибутима. Резултати добијени регресионом анализом говоре како се вредност зависне променљиве мења када се промени вредност једне независне променљиве, док су вредности осталих независних променљивих фиксиране. Основни задатак регресионе анализе је

апроксимација регресионе функције којом се представља веза између зависне и независних променљивих. Регресиона анализа се такође користи за оцењивање функционалне зависности између зависне и независних променљивих, као и природе те зависности. Сурогат моделирање се најчешће формулише као регресиони проблем.

Код ненадгледаног учења нема потребе за обележавањем података. Циљ је открити скривену структуру, или образац, у необележеним подацима. Типични примери ненадгледаног учења су кластеризација и асоцијација. Код кластеризације треба открити које групе унутар података постоје, нпр. које групе животиња постоје на основу неких карактеристика. Код проблема асоцијације треба открити правила као што су на пример: „Људи који купују X обично купују и Y “. Модели који користе и обележене и необележене податке спадају у групу полу-надгледаних метода.

Алгоритми који уче подстицањем интерагују са околином или окружењем и покушавају да максимизују награду или да минимизују казну. Награда долази споља, као последица понашања модела. Модел који учи на овај начин се често назива и агентом. Постоје и модели који поред спољашње награде користе и унутрашњу (на пример може се сматрати да агент има неку дозу радозналости).

4.1 Препроцесирање података

Након прикупљања података, а пре обуке модела, често је потребно процесирати податке, да би они били погоднији за алгоритам учења. Препроцесирање података може обухватати пречишћавање, трансформацију и редукцију података. Подаци могу имати нерелевантне делове или неки делови могу да недостају. Када део података недостаје у неким редовима, ти редови се могу игнорисати у потпуности или се могу допунити вредности које недостају. Вредности које недостају се допуњавају средњом вредношћу, најчешћом вредношћу, највероватнијом вредношћу или слично.

Често је за алгоритам обуке погодно да подаци буду нормализовани. Размотримо пример куповине куће. Део улазних података тада може да обухвата старост куће у годинама, број соба и цену. Алгоритам учења треба да научи да постоји велика разлика између куће са две собе и куће са двадесет соба. Опсег година и цена је пуно већи него опсег соба. Цене кућа могу да се разликују у хиљадама евра, а старост кућа се може разликовати и за 100 година или више. На основу ових опсега, алгоритам за учење може да сматра број соба небитним у односу на цену и старост. Да би утицај свих улазних атрибута био подједнак они се обично скалирају на исти опсег. Постоји више типова нормализације података, а неке од њих су мин-макс нормализација и z-скор стандардизација.

Мин-макс нормализација је линеарна трансформација. Нека су X оригинални подаци са минимумом \min_x и максимумом \max_x . Са опсега (\min_x, \max_x) подаци X се скалирају на нов задат опсег (\min_{new}, \max_{new}) коришћењем формуле:

$$X_{new} = \min_{new} + \frac{X - \min_x}{\max_x - \min_x} (\max_{new} - \min_{new}) \quad (4.1)$$

Најчешће се подаци скалирају на опсег $(0, 1)$. Мин-макс нормализација чува оригиналну зависност података. Максимум и минимум података су познати за прикупљени узорак, али некад нису познати за целу популацију. Овакав приступ има ману да може доћи до грешке ако се појави нова вредност која излази ван опсега узорка.

Z-скор стандардизација подразумева нормализацију података X , са узорцима x_1, \dots, x_N , одузимањем средње вредности $\bar{X} = \frac{1}{N} \sum_{i=1}^N x_i$ и скалирањем разлике помоћу стандардне девијације $\sigma_X = \sqrt{\frac{1}{N} \sum_{i=1}^N (x_i - \bar{X})^2}$:

$$X_{new} = \frac{x - \bar{X}}{\sigma_X} \quad (4.2)$$

Овај приступ је користан када минимум и максимум нису познати. Такође овај приступ може да буде користан када постоје подаци који доста одступају од већег дела узорка и тиме доминирају приликом мин-макс нормализације.

Редуковање података је скуп техника које се користе када је количина података превелика. Анализа је тежа када је количина података већа, па су тада ове технике неопходне. Једна од техника је селекција атрибута, где се чувају само улазне променљиве које су неопходне за креирање модела, а све остале се одбацују. Некада се прибегава трансформацијама које другачије кодирају податке, тако да се димензија података смањује. Ове трансформације могу да буду такве да је оригиналне податке могуће реконструисати или могу да буду такве да се након трансформације оригинални подаци не могу реконструисати.

Процес сурогат моделирања, почиње дизајнирањем експеримената, односно одабиром вредности улазних параметара, које ће бити коришћене за формирање почетног скупа података за обуку модела. Одабир улазних података и дизајн експеримената су важан део процеса сурогат моделирања. Након одабира улазних параметара, потребно је покренути рачунарске симулације и из њих прикупити податке за обуку сурогат модела. Следећи кораци су препроцесирање података, конструкција сурогат модела и његово обучавање коришћењем прикупљених података. У општем случају је тешко или немогуће унапред одредити укупан број узорака, који је потребан за креирање довољно прецизног сурогат модела. Из тог разлога, скуп података се обогаћује, одабиром нових узорака и покретањем симулација. Након обогаћивања скупа података, сурогат модел се поново обучава. Процес наизменичног обогаћивања скупа података и обучавања модела се понавља све док се не постигне задовољавајућа сличност са оригиналним моделом, који треба заменити.

5 Вештачке неуронске мреже

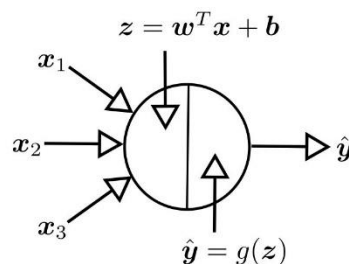
Вештачке неуронске мреже (енгл. *Artificial Neural Networks* - ANN) су по структури, функцији и обради информација сличне биолошким неуронским мрежама, али се ради о вештачким творевинама. Неуронска мрежа, у рачунарским наукама, представља повезану мрежу елемената који обрађују податке. Једну процесну јединицу неуронске мреже називамо неуроном. У овом поглављу ће бити описан модел вештачког неурона, затим ће бити описан најједноставнији облик дубоке неуронске мреже, који се назива вишеслојни перцептрон (енгл. *Multilayer Perceptron* или скраћено MLP), па ће бити дати сложенији типови неуронских мрежа, рекурентне и конволуционе мреже. Неуронске мреже се примењују на широк спектар проблема. Поред регресије, неуронске мреже налазе примену у класификацији података, кластеризацији, детектовању аномалија, машинском превођењу, транскрипцији, синтетисању и узорковању података, сегментацији слика итд.

5.1 Перцептрон

Модел вештачког неурона дат је на слици 5.1. Неуронска мрежа која се састоји од једног неурона се често назива перцептрон. Перцептрон је линеарни класификатор, који функционише на следећи начин: (1) Сви улази x_i се множе одговарајућим тежинским коефицијентима w_i , (2) Добијене вредности се сабирају и додаје се слободни коефицијент b (*bias*):

$$z = w_1 x_1 + w_2 x_2 + w_3 x_3 + \dots + w_n x_n + b = \mathbf{w}^T \mathbf{x} + b \quad (5.1)$$

(3) На добијену суму се примењује функција активације. Функција активације се другачије назива функцијом трансфера. Постоје различити типови ових функција, али се угрубо могу поделити на линеарне и нелинерне.



Слика 5.1 Модел вештачког неурона

Код вештачких неуронских мрежа користе се разни алгоритми за обучавање, којима се тежински коефицијенти подешавају на основу улазних података и жељеног излаза. Најчешће се обука мрежа своди на минимизацију функције губитка која представља разлику између излаза који мрежа даје и жељеног излаза. Алгоритми за минимизацију функције губитка, или циљне функције, као и детаљи обучавања неуронских мрежа ће бити обрађени касније. За сада је довољно рећи да ови алгоритми захтевају налажење парцијалног извода функције циља по тежинским коефицијентима мреже и да се тежински коефицијенти ажурирају на основу вредности извода. Скуп података који пролази кроз мрежу у току обуке, може бити веома велики, па се дели на мање целине

(енгл. *batch*). Пролазак једног делића података кроз мрежу, је једна итерација алгоритма за обуку. Један корак алгоритма за обуку се назива епоха. Једна епоха се завршава након што сви подаци за обуку прођу кроз мрежу. Број итерација у једној епохи зависи од величине дела података који пролази кроз мрежу (енгл. *batch size*) и величине скупа података за обуку. Код неуронских мрежа често је потребан велики број епоха пре него што вредност функције губитка падне на довољно малу вредност, односно пре него што алгоритам конвергира.

5.2 Функције трансфера

У овој секцији ће бити описане неке од често коришћених функција активације односно функција трансфера: сигмоидална функција (енгл. *sigmoid*), хиперболична тангентна функција (енгл. *tanh*), исправљена линеарна јединица (енгл. *Rectified Linear Unit - ReLU*) и скалирана експоненцијална линеарна јединица (енгл. *Scaled Exponential Linear Unit - SeLU*).

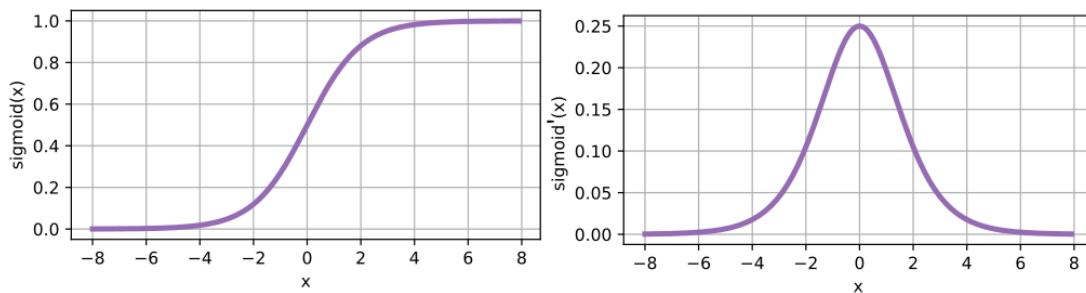
Сигмоидална функција је логистичка функција, што значи да без обзира на опсег улаза, излаз функције ће бити у опсегу између нуле и јединице. Код првих неуронских мрежа, научници су били заинтересовани за моделирање биолошких неурона који или шаљу сигнал или не шаљу сигнал, па је најпре као функција трансфера била коришћена функција прага. Функција прага даје вредност 0, када је улазна вредност мања од задатог прага, а у супротном функција има вредност 1. Међутим, пошто модерни алгоритми учења захтевају налажење извода односно градијената, прешло се на сигмоидалну функцију, јер је она глатка и представља диференцијабилну апроксимацију функције прага. Ова функција се често користи у излазном слоју приликом решавања проблема бинарне класификације, јер тада излаз неуронске мреже може бити интерпретиран као вероватноћа да дати улаз припада некој класи. Формула сигмоидалне функције је:

$$\text{sigmoid}(x) = \sigma(x) = \frac{1}{1+e^{-x}} \quad (5.2)$$

Први извод сигмоидалне функције је:

$$\text{sigmoid}'(x) = \sigma'(x) = \sigma(x)(1 - \sigma(x)) \quad (5.3)$$

Графички приказ сигмоидалне функције и њеног првог извода дат је на слици 5.2. Једна од мана сигмоидалне функције је да њен градијент нестаје за велике позитивне или велике негативне вредности.



Слика 5.2 Сигмоидална функција и њен први извод

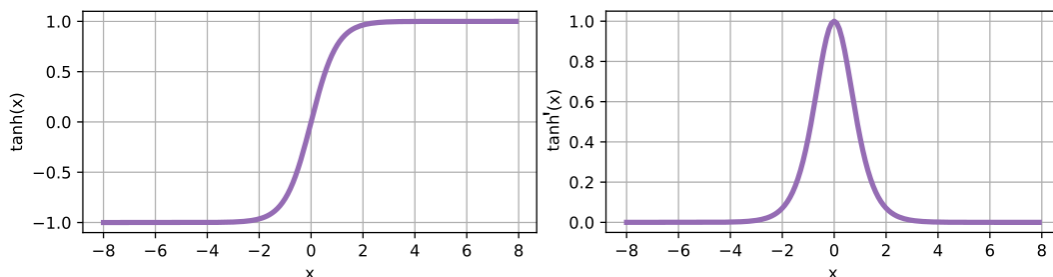
Слично као сигмоидална функција, и хиперболична тангентна функција трансформише улаз на неки опсег, с тим што је тај опсег у овом случају $(-1, 1)$. Формула хиперболичне тангентне функције је:

$$\tanh(x) = \frac{2}{1+e^{-2x}} - 1 \quad (5.4)$$

а њен први извод је:

$$\tanh'(x) = 1 - \tanh(x)^2 \quad (5.5)$$

Графички приказ ове функције и њеног извода је дат на слици 5.3. Може се приметити да се хиперболична тангентна функција понаша као линеарна функција у околини нуле. Облик функције је сличан као сигмоидални али ова функција је симетрична у односу на координатни почетак.

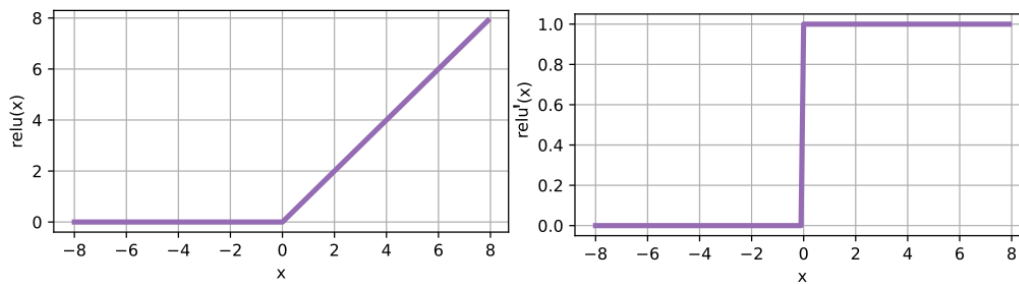


Слика 5.3 Хиперболична тангентна функција и њен први извод

Једна од најпопуларнијих активационих функција је исправљена линеарна функција, која је веома једноставна за имплементацију а често даје добре перформансе. Ова функција је дефинисана као:

$$\text{relu}(x) = \max(x, 0) \quad (5.6)$$

Графички приказ ReLU функције и њеног првог извода је дат на слици 5.4. Ова функција задржава само вредности позитивних улаза, а занемарује негативне. Када је улаз негативан извод функције је нула, а када је улаз позитиван извод је један. Када је улаз једнак тачно нули, ова функција није диференцијабилна. У овом случају усваја се да је извод једнак нули. Разлог за честу употребу ове функције је да се њен извод добро понаша, и да је проблем нестајања градијената смањен код ове функције у односу на сигмоидалну и хиперболичну тангентну функцију.



Слика 5.4 Исправљена линеарна функција и њен први извод

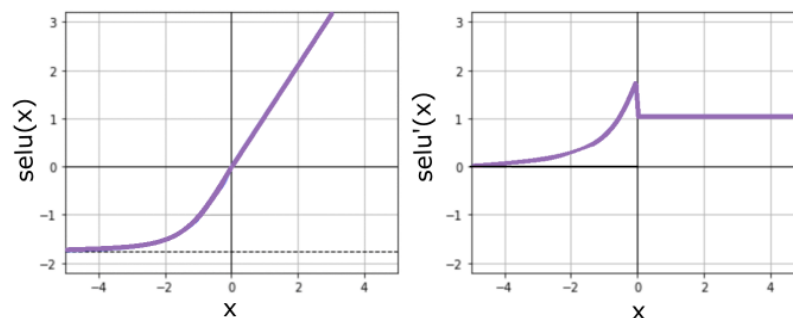
Скалабилна експоненцијална линеарна јединица (SeLU) је функција активације која истовремено нормализује коефицијенте мреже. Под нормализацијом сматра се одузимање средње вредности и дељење са стандардном девијацијом тако да компоненте мреже имају средњу вредност нула и стандардну девијацију један. Овај тип нормализације, се назива интерном нормализацијом јер се нормализација врши директно унутар неурона мреже. Формула SeLU функције је:

$$selu(x) = \lambda \begin{cases} x & , x > 0 \\ \alpha e^x - \alpha & , x \leq 0 \end{cases} \quad (5.7)$$

а њен први извод је:

$$selu'(x) = \lambda \begin{cases} 1 & , x > 0 \\ \alpha e^x & , x \leq 0 \end{cases} \quad (5.8)$$

Графички приказ је дат на слици 5.5.

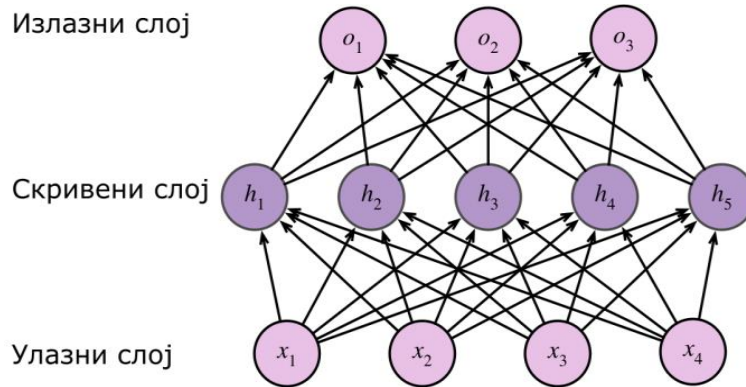


Слика 5.5 Скалирана експоненцијална линеарна јединица и њен први извод

5.3 Вишеслојни перцептрон

Вишеслојни перцептрон се састоји од више међусобно повезаних слојева неурона. Слој који нема претходника се назива улазним слојем, док се слој који нема следбеника назива излазним слојем. Сваки слој који има претходника је потпуно повезан са њим и од њега добија улаз. Такође, сваки слој који има следбеника је потпуно повезан са њим и њему шаље сигнал. Сигнал путује кроз вишеслојни перцептрон од улазног слоја, ка излазном, у току процеса који се назива пропација унапред. Мотивација за креирање вишеслојног перцептрона је превазилажење ограничења обичног перцептрона. Перцептрон може да класификује само линеарно сепарабилне касе. Линеарно сепарабилни проблеми класификације су они проблеми код којих је класе могуће

одвојити једном хипер-равни. Вишеслојни перцептрон омогућава решавање проблема класификације и у случајевима када класе нису линеарно сепарабилне. На слици 5.6 је приказан вишеслојни перцептрон са 4 улаза, 3 излазна и једним скривеним слојем који има 5 неурона. Неуроне унутар слоја често називамо и јединицама. Конструкција дубоке неуронске мреже подразумева одабир хиперпараметара. Под одабиром хиперпараметара се сматра дизајн топологије мреже, односно одабир броја слојева и броја неурона у сваком од слојева, затим одабир алгоритма за обуку и његових параметара итд. Укратко, хиперпараметрима се дефинише структура неуронске мреже и начин на који се она обучава.



Слика 5.6 Вишеслојни перцептрон са једним скривеним слојем [44]

Нека је $\mathbf{X} \in \mathbb{R}^{n \times d}$ део улаза у неуронску мрежу, при чему је n број узорака, d број улазних атрибута и q број излаза из мреже [44, 45]. Нека је $\mathbf{H} \in \mathbb{R}^{n \times h}$ излаз скривеног слоја вишеслојног перцептрона са једним скривеним слојем од h скривених неурона или скривених јединица. Пошто су скривени и излани слој потпуно повезани, имамо да су тежински коефицијенти скривеног слоја $\mathbf{W}^{(1)} \in \mathbb{R}^{d \times h}$ а $\mathbf{b}^{(1)} \in \mathbb{R}^{1 \times h}$ слободни коефицијенти скривеног слоја, и $\mathbf{W}^{(2)} \in \mathbb{R}^{h \times q}$ тежински коефицијенти излазног слоја а $\mathbf{b}^{(2)} \in \mathbb{R}^{1 \times q}$ слободни коефицијенти излазног слоја. Излаз из мреже $\mathbf{O} \in \mathbb{R}^{n \times q}$ је тада дефинисан као:

$$\begin{aligned} \mathbf{H} &= \phi(\mathbf{X}\mathbf{W}^{(1)} + \mathbf{b}^{(1)}) \\ \mathbf{O} &= \mathbf{X}\mathbf{W}^{(2)} + \mathbf{b}^{(2)} \end{aligned} \quad (5.9)$$

при чему је ϕ одабрана активациона функција. Зарад креирања експресивнијих мрежа можемо додати још скривених слојева, нпр. излаз из мреже са два скривена слоја би био дефинисан са: $\mathbf{H}_1 = \phi_1(\mathbf{X}\mathbf{W}^{(1)} + \mathbf{b}^{(1)})$, $\mathbf{H}_2 = \phi_2(\mathbf{X}\mathbf{W}^{(2)} + \mathbf{b}^{(2)})$ и $\mathbf{O} = \mathbf{X}\mathbf{W}^{(3)} + \mathbf{b}^{(3)}$. Универзална теорема апроксимације говори да је са једним скривеним слојем, са довољним бројем неурона, могуће апроксимирати било коју континуалну функцију. Међутим, у пракси се чешће користе мреже које имају више слојева са мање неурона, такозване дубоке неуронске мреже, уместо плићих неуронских мрежа са веома великим бројем неурона у скривеном слоју. У току обуке неуронске мреже вршимо такозвану пропацију унапред и пропацију уназад. Машинско учење које укључује дубоке неуронске мреже се често назива дубоким учењем.

5.3.1 Пропагација унапред

Пропагација унапред се односи на прорачуне и чување међурезултата, приликом проласка података од улаза мреже ка излазу. Зарад једноставности посматраћемо мрежу са једним скривеним слојем који нема слободне коефицијенте. Нека је $\mathbf{x} \in \mathbb{R}^d$ узорак, тада имамо међурезултат:

$$\mathbf{z} = \mathbf{W}^{(1)} \mathbf{x}, \quad (5.10)$$

где су $\mathbf{W}^{(1)} \in \mathbb{R}^{h \times d}$ тежински коефицијенти скривеног слоја. Након проласка $\mathbf{z} \in \mathbb{R}^h$ кроз функцију трансфера ϕ добијамо излаз из скривеног слоја:

$$\mathbf{h} = \phi(\mathbf{z}) \quad (5.11)$$

Овај излаз $\mathbf{h} \in \mathbb{R}^h$ такође представља међурезултат. Нека излазни слој такође нема слободне коефицијенте, већ само тежинске $\mathbf{W}^{(2)} \in \mathbb{R}^{q \times h}$, тада је излаз из мреже:

$$\mathbf{o} = \mathbf{W}^{(2)} \mathbf{h} \quad (5.12)$$

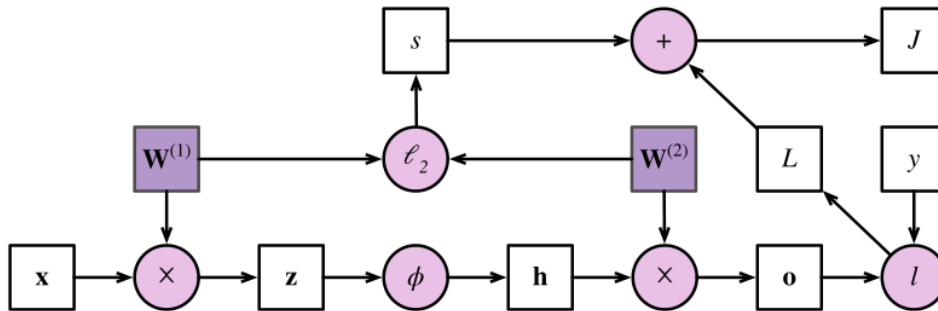
Нека је l функција губитка и нека је y жељени излаз из мреже, грешка мреже је тада дефинисана као:

$$L = l(\mathbf{o}, y) \quad (5.13)$$

Зарад ограничења вредности које тежински коефицијенти могу да добију можемо користити неку врсту регуларизације, омогућавајући стабилнију обуку мреже. Ако је l_2 регуларизација коју користимо са хиперпараметром λ и $s = \frac{\lambda}{2} (\|\mathbf{W}^{(1)}\|_F^2 + \|\mathbf{W}^{(2)}\|_F^2)$ тада можемо дефинисати регулисану функцију циља:

$$J = L + s \quad (5.14)$$

при чему је $\|\dots\|_F^2$ Фробенијусова норма матрице, односно l_2 норма примењена на матрицу која је претворена у вектор (ако је дат вектор (a_1, \dots, a_k) тада је његова l_2 норма $\sqrt{\sum_{i=1}^k a_i^2}$). На слици 5.7 је дат шематски приказ графа прорачуна пропагације унапред. Квадрати на слици 5.7 означавају променљиве, а кружићи означавају операторе.



Слика 5.7 Шематски приказ графа прорачуна пропагације унапред [44]

5.3.2 Пропагација уназад

Аутоматска диференцијација је допринела имплементацији и развоју разних алгоритама за обуку. Како подаци пролазе кроз мрежу, може се формирати граф прорачуна који прати како вредности у неуронској мрежи зависе једне од других. Да бисмо срачунали изводе, треба да се крећемо уназад кроз граф и да примењујемо ланчано правило извода. Некада је потребно урадити део прорачуна ван графа прорачуна. На пример може бити потребно срачунавање привремене променљиве за коју не желимо да рачунамо градијент. Тада се рачунски утицај одваја од коначног резултата. На пример нека имамо $z = x * u$ и $u = x * x$ и нека је потребно срачунати директни утицај x на z . У овом случају може се креирати нова променљива u која узима вредност u , али тако да детаљи срачунавања буду сакривени. Пошто u нема претходника у графу прорачуна, градијенти не теку кроз u до x . Рачунањем градијента $z = x * u$ се тада добија x уместо $3 * x * x$ (пошто је $z = x * x * x$).

Пропагација уназад се односи на рачунање градијената параметара неуронске мреже [46]. Под параметрима неуронске мреже мисли се на тежинске коефицијенте. Пропагацијом уназад пролазимо кроз мрежу уназад од излазног слоја ка улазном, и рачунамо изводе. Нека су дате функције $Y = f(X)$, $Z = g(Y)$, у којима су X, Y, Z тензори произвољних величина, ланчано правило извода Z по X даје:

$$\frac{\partial Z}{\partial X} = \Pi \left(\frac{\partial Z}{\partial Y}, \frac{\partial Y}{\partial X} \right) \quad (5.15)$$

Овде смо увели оператор Π који множи своје аргументе након неопходних операција као што су транспоновање и замена позиција улаза. За векторе ово је обично множење матрица, за тензоре вишег реда користи се одговарајући поступак, а нотација Π служи да сакрије детаље. Пошто су параметри мреже са једним скривеним слојем $\mathbf{W}^{(1)}$ и $\mathbf{W}^{(2)}$, пропагацијом уназад треба да добијемо градијенте $\partial J / \partial \mathbf{W}^{(1)}$ и $\partial J / \partial \mathbf{W}^{(2)}$. Први корак је срачунавање градијената функције циља J по L и s :

$$\frac{\partial J}{\partial L} = \frac{\partial J}{\partial s} = 1 \quad (5.16)$$

Након тога се рачуна градијент по излазу мреже, коришћењем ланчаног правила:

$$\frac{\partial J}{\partial \mathbf{o}} = \Pi \left(\frac{\partial J}{\partial L}, \frac{\partial L}{\partial \mathbf{o}} \right) = \frac{\partial J}{\partial \mathbf{o}} \in \mathbb{R}^q \quad (5.17)$$

Градијенти вредности регуларизације s по параметрима мреже су:

$$\frac{\partial s}{\partial \mathbf{w}^{(1)}} = \lambda \mathbf{W}^{(1)}, \frac{\partial s}{\partial \mathbf{w}^{(2)}} = \lambda \mathbf{W}^{(2)} \quad (5.18)$$

Градијент $\partial J / \partial \mathbf{W}^{(2)}$ је:

$$\frac{\partial J}{\partial \mathbf{w}^{(2)}} = \Pi \left(\frac{\partial J}{\partial \mathbf{o}}, \frac{\partial \mathbf{o}}{\partial \mathbf{w}^{(2)}} \right) + \Pi \left(\frac{\partial J}{\partial s}, \frac{\partial s}{\partial \mathbf{w}^{(2)}} \right) = \frac{\partial J}{\partial \mathbf{o}} \mathbf{h}^T + \lambda \mathbf{W}^{(2)} \quad (5.19)$$

Да бисмо добили градијенте по $\mathbf{W}^{(1)}$ треба да наставимо пропагацију уназад, па је градијент по излазу скривеног слоја $\partial J / \partial \mathbf{h} \in \mathbb{R}^h$:

$$\frac{\partial J}{\partial \mathbf{h}} = \Pi \left(\frac{\partial J}{\partial \mathbf{o}}, \frac{\partial \mathbf{o}}{\partial \mathbf{h}} \right) = \mathbf{W}^{(2)T} \frac{\partial J}{\partial \mathbf{o}} \quad (5.20)$$

Пошто се активациона функција ϕ примењује на елементе, рачунање градијента $\partial J / \partial \mathbf{z} \in \mathbb{R}^h$ захтева коришћење елементарног оператора производа \odot :

$$\frac{\partial J}{\partial \mathbf{z}} = \Pi \left(\frac{\partial J}{\partial \mathbf{h}}, \frac{\partial \mathbf{h}}{\partial \mathbf{z}} \right) = \frac{\partial J}{\partial \mathbf{h}} \odot \phi'(\mathbf{z}) \quad (5.21)$$

На крају рачунамо градијент:

$$\frac{\partial J}{\partial \mathbf{W}^{(1)}} = \Pi \left(\frac{\partial J}{\partial \mathbf{z}}, \frac{\partial \mathbf{z}}{\partial \mathbf{W}^{(1)}} \right) + \Pi \left(\frac{\partial J}{\partial s}, \frac{\partial s}{\partial \mathbf{W}^{(1)}} \right) = \frac{\partial J}{\partial \mathbf{z}} \mathbf{x}^T + \lambda \mathbf{W}^{(1)} \quad (5.22)$$

Приликом обуке мреже пропација унапред и уназад зависе једна од друге. Најпре пролазимо пропацијом унапред кроз мрежу и чувамо међурезултате, које затим користимо у пропацији уназад. Регуларизација тежина, у току пропације унапред, зависи од вредности $\mathbf{W}^{(1)}$ и $\mathbf{W}^{(2)}$, које се добијају оптимзационим алгоритмом спровођењем пропације уназад. Са друге стране, рачунање градијената зависи од излаза скривеног слоја, који добијамо пропацијом унапред итд. У току обуке најпре се иницијализују сви параметри, а затим се смењују пропација унапред и уназад, ажурирајући параметре модела. Чување међурезултата је један од разлога зашто је потребно више меморије за обуку мреже, него за обичну предикцију или евалуацију. Број међурезултата зависи од броја слојева, броја неурона и броја узорака који пролази кроз мрежу у једном кораку алгоритма обуке.

Многи алгоритми за обуку мреже захтевају да параметри (тежински и слободни коефицијенти) буду иницијализовани по некој унапред дефинисаној дистрибуцији. Одабир иницијализације игра значајну улогу у обучавању мреже и може бити кључан за одржавање нумеричке стабилности. Одабир може бити повезан и са одабиром функције трансфера. Одабир функције трансфера и начина иницијализације параметара може да утиче на конвергенцију алгоритма за обуку. Лош одабир иницијализације може довести до проблема са нестајућим и експлодирајућим градијентима у току обуке. Нека имамо неуронску мрежу са слојева L , улазом \mathbf{x} и излазом \mathbf{y} . За сваки слој имамо функцију трансфера ϕ_l и тежинске коефицијенте $\mathbf{W}^{(l)}$, а излази скривеног слоја су $\mathbf{h}^{(l)}$ за $l = 1, \dots, L$ при чему је $\mathbf{h}^{(0)} = \mathbf{x}$. Мрежа може бити формулисана као:

$$\mathbf{h}^{(l)} = \phi_l(\mathbf{h}^{(l-1)}) \quad \text{и} \quad \mathbf{o} = \phi_L \circ \dots \circ \phi_1(\mathbf{x}) \quad (5.23)$$

Ако су сви излази скривених слојева, као и сви улази, вектори, тада градијенте излаза по параметрима $\mathbf{W}^{(l)}$ можемо записати као:

$$\frac{\partial \mathbf{o}}{\partial \mathbf{W}^{(l)}} = \frac{\partial \mathbf{h}^{(L)}}{\partial \mathbf{h}^{(L-1)}} \dots \frac{\partial \mathbf{h}^{(l+1)}}{\partial \mathbf{h}^{(l)}} \frac{\partial \mathbf{h}^{(l)}}{\partial \mathbf{W}^{(l)}} \quad (5.24)$$

Другим речима градијент је производ $L - l$ матрица $\frac{\partial \mathbf{h}^{(L)}}{\partial \mathbf{h}^{(L-1)}} \dots \frac{\partial \mathbf{h}^{(l+1)}}{\partial \mathbf{h}^{(l)}}$ и вектора $\frac{\partial \mathbf{h}^{(l)}}{\partial \mathbf{W}^{(l)}}$. Иницијалне матрице могу имати широк опсег сопствених вредности, које могу бити мале или велике, а самим тим и њихови производи могу бити веома мали или веома

велики. Градијенти непредвидивих величина доводе до нестабилности алгоритама за оптимизацију. Можемо ажурирати параметре превеликим (проблем експлодирајућих градијената) или премалим вредностима (проблем нестајућих градијената). На пример, уколико користимо сигмоидалну функцију, чији градијенти имају малу вредност када су улази велики или мали, и ако вршимо пропацију уназад кроз велики број слојева, велика је вероватноћа да ће градијент бити одсечен код неког од слојева. Поред пажљивог одабира иницијализације, постоје и различите методе за регуларизацију, које могу да допринесу умањењу или потпуном избегавању проблема са нестајућим и експлодирајућим градијентима. Ове методе ће бити описане касније.

5.3.3 Иницијализација параметара

Да бисмо показали важност иницијализације, претпоставимо да сви тежински коефицијенти мреже имају исту иницијалну вредност. У том случају, у току пропације унапред, сваки неурон унутар једног слоја добија исти улаз и даје исти излаз, а у току пропације уназад добићемо исте градијенте за све неуроне у слоју. Уколико користимо детерминистички алгоритам за обуку мреже, нећемо моћи да се ослободимо ове симетрије и цео слој мреже се може свести на један неурон, што није жељено понашање. Да бисмо искористили неуроне унутар слоја на прави начин, сваки неурон треба да има другачије коефицијенте, тиме моделирајући различит део проблема који се решава.

Један од начина за иницијализацију параметара је додељивање насумичне вредности из нормалне дистрибуције сваком тежинском коефицијенту у мрежи. Овај начин иницијализације може да се покаже добрим у пракси за проблеме умерене величине. Међутим овај тип иницијализације није добар за веће мреже, јер варијанса тежинских коефицијената тада није довољна, што може да доведе до проблема са градијентима. Због овога се уводе други типови иницијализације, које узимају у обзир број неурона који се налази у слоју мреже. У овој подсекцији ћемо укратко описати неке типове иницијализација које се често користе: Гзавије иницијализацију, ортогоналну иницијализацију и Хе иницијализацију. Слободни коефицијенти се најчешће постављају на нулу на почетку обуке, мада се и на њих могу применити разни типови иницијализације.

Нека су o_i излази потпуно повезаног слоја мреже, без нелинеарности. Са n_{in} улаза, обележених са x_j и одговарајућим тежинским коефицијентима w_{ij} , излаз је тада дат са $o_i = \sum_{j=1}^{n_{in}} w_{ij} x_j$. Тежински коефицијенти су одабрани независно, из исте дистрибуције. Нека ова дистрибуција има средњу вредност једнаку нули и варијансу σ^2 . Нека улази имају средњу вредност нула и варијансу γ^2 , и нека су улази међусобно независни и независни од тежинских коефицијената. Средња вредност излаза је тада:

$$E[o_j] = \sum_{j=1}^{n_{in}} E[w_{ij} x_j] = \sum_{j=1}^{n_{in}} E[w_{ij}] E[x_j] = 0 \quad (5.25)$$

а варијанса је:

$$\text{Var}[o_j] = E[o_j^2] - (E[o_j])^2 = \sum_{j=1}^{n_{in}} E[w_{ij}^2]E[x_j^2] - 0 = n_{in}\sigma^2\gamma^2 \quad (5.26)$$

Један од начина да варијанса буде фиксна је да важи $n_{in}\sigma^2 = 1$. Узмимо сада у обзир пропагацију уназад. Уколико не важи $n_{out}\sigma^2 = 1$, где је n_{out} број излаза из слоја, варијанса градијената може да постане велика. Дакле треба да важи $\frac{1}{2}(n_{in} + n_{out})\sigma^2 = 1$, односно $\sigma = \sqrt{\frac{2}{n_{in}+n_{out}}}$. Гзавијеова иницијализација обично користи вредности из нормалне дистрибуције са средњом вредношћу 0 и варијансом $\sigma^2 = \frac{2}{n_{in}+n_{out}}$. Да би се Гзавијеова иницијализација применила на униформну дистрибуцију, треба приметити да униформна дистрибуција $U(-a, a)$ има варијансу $a^2/3$, заменом ове вредности у $\sigma^2 = \frac{2}{n_{in}+n_{out}}$ добијамо:

$$U\left(-\sqrt{\frac{6}{n_{in}+n_{out}}}, \sqrt{\frac{6}{n_{in}+n_{out}}}\right) \quad (5.27)$$

Овај тип иницијализације се често назива и Глорот иницијализација [47]. Иако приликом извођења постоји претпоставка да су активације линеарне, овај тип иницијализације често добро функционише у пракси и када су активације нелинеарне.

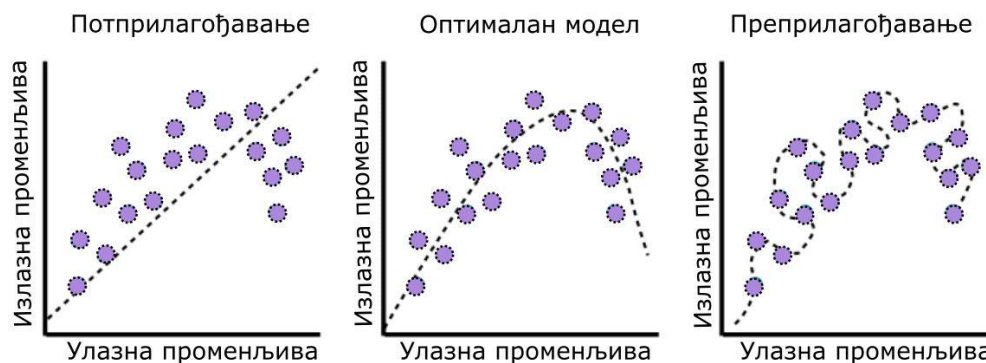
Ортогонална иницијализација најпре иницијализује тежинске коефицијенте вредностима из нормалне дистрибуције а затим се врши декомпозиција матрице тежинских коефицијената тако да, на крају, резултујућа матрица буде ортогонална. Матрица је ортогонална ако су њене колоне и редови ортогонални вектори, односно ако је њихов скаларни производ једнак нули. Овај тип иницијализације се посебно користи код рекурентних мрежа за иницијализацију рекурентних тежинских коефицијената. Код рекурентних мрежа имамо посебно дугачке ланце множења матрица. Ортогоналне матрице имају ту особину да је производ двеју ортогоналних матрица, такође ортагонална матрица, па овај тип иницијализације помаже да обука остане нумерички стабилна, умањујући могућност настајања проблема нестајућих и експлодирајућих градијената.

Код проблема класификације слика, Гзавијеова иницијализација није давала значајна побољшања у пару са ReLU активацијом, па је изведена тзв. Хе иницијализација, која иницијализује тежинске коефицијенте вредностима из нормалне дистрибуције $N(0, \sqrt{2/n_i})$, при чему је n_i број јединица у слоју.

5.4 Проблем генерализације

Приликом обучавања неуронских мрежа, често користимо податке. Скуп података делимо на скуп података за обуку мреже, и скуп података за тестирање мреже. Додатно, некад користимо и скуп података за валидацију мреже, који не користимо директно у току обуке, али га користимо за анализирање перформанси модела, пре тестирања. Добре предикције на скупу за обуку су само успутни циљ учења. Прави циљ учења је што приближнија апроксимација праве функције и, последично, прецизне предикције

на скупу података који није коришћен у току обуке неуронске мреже. Под генерализацијом сматрамо способност модела да се прилагоди и реагује на нове податке [48], који нису коришћени у току учења али су из исте дистрибуције као подаци, који су коришћени. Уколико је модел превише добро научио да врши предикције на подацима за обуку, он може да прави лоше предикције када се употребе нови подаци. Овај проблем је познат као преучавање или преприлагођавање (енгл. *overfitting*). Када модел врши добре предикције на скупу за обуку често се каже и да модел добро фитује скуп података за обуку. Супротно томе, потприлагођавање, (енгл. *underfitting*) такође може да се јави. Модел може да не фитује ни скуп података за обуку, и да последично врши лоше предикције на скупу за тестирање. Идеално би било одабрати модел који стоји тачно између ова два проблема, модел који добро фитује скуп података за обуку, али не превише добро, и који даје добре предикције на тестним подацима. На слици 5.8 је дат приказ модела са потприлагођавањем, оптималног модела и модела са преприлагођавањем.



Слика 5.8 Приказ модела са потприлагођавањем, оптималним моделом и модела са преприлагођавањем

Теорија и примена дубоких неуронских мрежа се брзо развијају, али у општем случају још увек није потпуно јасно како, када и зашто мреже успевају да генерализују добро. Разлику између перформанси модела на скупу за обуку и скупу за тестирање називамо **генерализациони јаз**. Када је овај јаз велики, кажемо да је модел преучио. Често се тада сматра да је модел који смо обучили превише комплексан и да можда треба смањити број улазних атрибута, број параметара мреже и слично. Ако је наш модел веома експресиван, и савршено фитује сваки узорак из скупа за обуку, чак и када је тај скуп веома велики, можемо да ограничимо опсег вредности које параметри могу да имају, или увести неки тип регуларизације. Међутим, некад иако је грешка на скупу за обуку веома мала, генерализациони јаз се може смањити додавањем слојева, неурона или дужом обуком мреже. Још интересантније, некад се почетним повећавањем комплексности модела најпре добија лошија генерализација, али како се модел још више усложњава генерализација постаје боља. Ово говори да генерализација модела не мора монотонно да зависи од његове комплексности.

5.4.1 Типови грешака

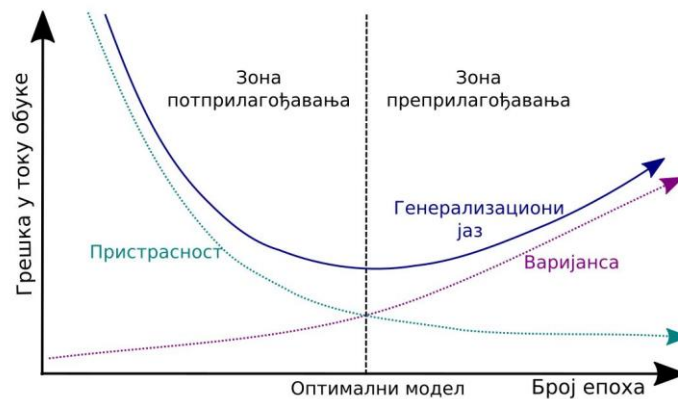
У општем случају грешке које модел прави у току учења се могу поделити на: (1) пристрасност (енгл. *bias*), (2) грешку варијансе и (3) грешку коју није могуће редуковати без обзира на алгоритам који се користи. Пристрасност је систематска грешка и односи се на претпоставке које модел прави. То су претпоставке које упрошћавају проблем. Низак степен пристрасности означава да модел прави мање претпоставки током учења, док висок степен пристрасности означава да је модел направио велики број претпоставки током учења. Линеарни алгоритми за предвиђање, као што су линеарна и логистичка регресија, имају високу пристрасност, лакши су за разумевање, али су мање флексибилни. Ниску пристрасност имају стабла одлуке, метода потпорних вектора итд. Грешка варијансе се односи на промену апроксимације са променом скупа за обуку. Идеално, варијанса модела не би требала да буде велика приликом промене скупа за обуку. Алгоритми који имају висок степен варијансе су осетљиви на специфичности података који се користе за обуку. Ниска варијанса модела означава да ће мале промене у подацима за обуку, дати мале измене у апроксимираној функцији. Малу варијансу имају једноставнији модели учења, као што је на пример линеарна регресија. Висока варијанса модела означава да ће мале промене у подацима за обуку, изазвати велике промене у апроксимираној функцији. Генерално, нелинеарни алгоритми, као што су на пример стабла одлуке, који имају већи степен флексибилности имају и велику варијансу. Циљ сваког алгоритма машинског учења је да има ниску системску грешку и ниску варијансу. Наравно, смањењем пристрасности, повећава се варијанса, а смањењем варијансе се повећава пристрасност, па је потребно наћи компромисно решење. У стварности је немогуће тачно срачунати пристрасност и варијансу, јер функција коју треба апроксимирати често није унапред позната. Без обзира на то, пристрасност и варијанса као концепти нам омогућавају да боље разумемо понашање алгоритама машинског учења.

5.4.2 Рано заустављање

Пошто неуронске мреже могу да имају веома велики број параметара, њима можемо интерполирати скуп података за обуку, фитујући га савршено. Међутим, ово савршено фитовање се може остварити тек након великог броја корака алгоритма за учење. Једна од метода која потенцијално омогућава бољу генерализацију модела се назива раним заустављањем (енгл. *early stopping*). Уместо ограничавања вредности параметара мреже, можемо користити валидациони скуп података и у току обуке посматрати перформансе модела на скупу за обуку и на скупу за валидацију. Уколико се грешка предвиђања на скупу за валидацију није значајно смањила у последњих e корака алгоритма за обуку, онда можемо прекинути обуку. Параметар e се назива прагом стрпљења (енгл. *patience criteria*).

На слици 5.9 је дат шематски приказ раног заустављања. Код неуронских мрежа, у току обуке варијанса модела се повећава, а степен пристрасности опада. Оптимални модел је онај модел који се налази на пресеку ове две грешке, и такав модел добијамо ако обуку зауставимо у тренутку када се грешка на скупу података за валидацију не смањује, односно када генерализациони јаз почиње да расте.

Поред омогућавања боље генерализације, предност методе раног заустављања је и уштеда времена, јер се обука завршава раније. Када не постоји шум у подацима, ова метода може да не доведе до значајнијих побољшања генерализације. Са друге стране, ако постоји шум у подацима (грешке у мерењу, лоше обележене класе итд.) ова метода може бити кључна јер спречава мрежу да интерполира лоше податке са шумом.



Слика 5.9 Шематски приказ раног заустављања

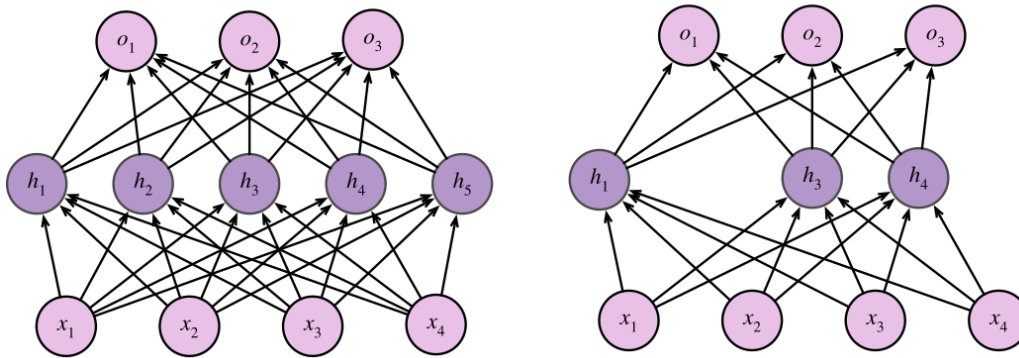
5.4.3 Смањивање тежинских коефицијената

Друга метода за побољшање степена генерализације модела је смањивање тежинских коефицијената (енгл. *weight decay*). Ова метода се састоји у томе да се додаје регуларизација на функцију губитка којом се кажњавају велике вредности тежинских коефицијената. Често се овај тип регуларизације користи уз рано заустављање.

5.4.4 Регулација изостављањем неурона

Један од начина да се изрази једноставност неког модела је неосетљивост на мале измене у улазу. Ова неосетљивост на мале измене значи да је функција, апроксимирана неуронском мрежом, глатка. На пример ако желимо да класификујемо слике, насумични шум настао у пар пиксела на слици не би требало да прави икакав проблем приликом класификације. Регулација изостављањем неурона (енгл. *dropout*) подразумева уношење шума унутар сваког слоја мреже у току пропагације унапред, искључивањем односно изостављањем неких неурона [49-52]. У току обуке неурони из једног слоја могу да створе међузависност са неуронима из претходног слоја ослањајући се на шаблоне активације. Ово се назива коадаптацијом. Регулација изостављањем неурона разбија овај вид зависности и омогућава неуронима да боље генерализују. Са вероватноћом изостављања p , одговарајући удео неурона из слоја се изоставља. Приликом рачунања излаза из скривеног слоја, он се нормализује да би се надокнадило изостављање неурона, тако да је очекивана вредност излаза скривеног слоја непромењена. Шематски приказ регуларизације изостављањем неурона је дат на слици 5.10. Услед изостављања, нестају неурони h_2, h_5 из скривеног слоја. Последице излаза из слоја више не зависи од изостављених неурона, а градијенти који зависе од њих нестају у току пропагације уназад. У току обуке у сваком кораку неурони насумично нестају, тако да излаз не може да постане превише завиштан ни од једног неурона h_1, \dots, h_5 .

а) Пре регуларизације изостављањем б) После регуларизације изостављањем



Слика 5.10 Шематски приказ регуларизације изостављањем неурона

У току тестирања, регуларизација изостављањем неурона се обично искључује. Након обуке модела, не изостављају се неурони. Међутим, има и изузетака, неки истраживачи користе технике изостављања и приликом тестирања модела да би проценили неизвесност предикција неуронске мреже. Ако се предикција поклапа, приликом неколико случаја са изостављањем неурона, можемо да сматрамо да је таква предикција поузданија.

5.5 Функције губитка

Функција губитка је хиперпараметар мреже, што значи да приликом конструкције мреже треба да одаберемо функцију губитка коју ћемо користити. Уопштено говорећи функције губитка могу бити подељене на оне које се користе за проблеме класификације и оне које се користе за проблеме регресије. Пошто се сурогат моделирање дефинише као регресиони проблем, у наставку ће бити дате неке од често коришћених функција губитка за овај тип проблема.

5.5.1 Функције губитка за проблеме регресије

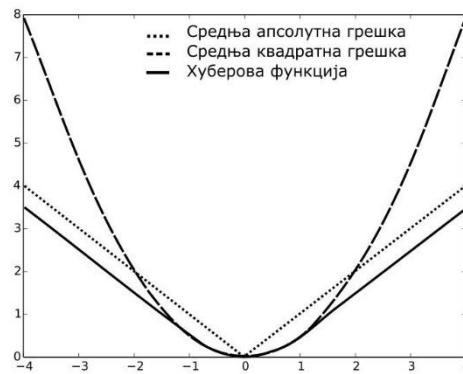
Средња квадратна грешка је функција губитка која се најчешће користи за регресионе проблеме, а њена формула је:

$$L_{mse} = \frac{1}{N} \sum_{i=1}^N (\hat{y}_i - y_i)^2 \quad (5.28)$$

при чему је N број узорака, y жељена односно права вредност, а \hat{y} предвиђена вредност. Ова функција пенализује модел који праве велике грешке, јер су грешке квадриране. Ова особина чини функцију мање робуством када постоје вредности које доста одступају. Друга најчешће коришћена функција губитка за регресионе проблеме је средња апсолутна грешка. Формула ове функције је дата са:

$$L_{mae} = \frac{1}{N} \sum_{i=1}^N |\hat{y}_i - y_i| \quad (5.29)$$

Ова функција је робустнија од средње квадратне грешке, али није глатка. Док је средња квадратна грешка континуално диференцијабилна, средња апсолутна није (прекид у нули). Ова функција је тежа за оптимизацију.



Слика 5.11 Средња апсолутна грешка, средња квадратна грешка и Хуберова функција губитка

Функција која комбинује својства претходне две функције је Хуберова функција. Ова функција се приближава вредности средње квадратне грешке када је разлика између жељених и добијених вредности мања од параметра δ , а када је разлика већа, ова функција се понаша као средња апсолутна грешка:

$$L_{huber}(y, \hat{y}) = \begin{cases} \frac{1}{2}(y - \hat{y})^2, & |y - \hat{y}| \leq \delta \\ \delta|y - \hat{y}| - \frac{1}{2}\delta^2, & |y - \hat{y}| > \delta \end{cases} \quad (5.30)$$

где је y жељена односно права вредност, а \hat{y} предвиђена вредност. Псеудо Хуберова функција, чија је формула $\delta^2(\sqrt{1 + (y - \hat{y})^2/\delta^2} - 1)$, је глатка апроксимација Хуберове функције. Постоје и друге глатке апроксимације Хуберове функције, али се претходно наведена најчешће користи. Ова функција је погоднија за оптимизацију од средње апсолутне грешке, и континуално је диференцијабилна. Такође, Хуберова функција је мање осетљива на вредности које доста одступају, па се зато често користи у регресионим проблемима који захтевају да функција губитка буде робусна. Једна од мана ове функције је што уводи додатни параметар δ који је потребно одредити. На слици 5.11 су приказане све три функције губитка за регресионе проблеме.

5.6 Евалуација модела

Иако нам вредност функције губитка у току обуке даје неку оцену грешке на скупу података за обуку, ово није довољно. Након обуке модела, тестирају се перформансе модела, најчешће на тестном скупу. Функције губитка се могу користити и као метрике за процену перформанси, али се најчешће не користи иста функција за евалуацију модела. Као и функције губитка, и метрике за евалуацију модела се могу поделити на оне које се користе за регресионе проблеме и оне које се користе за класификационе проблеме.

5.6.1 Функције за евалуацију регресионих проблема

Корен средње квадратне грешке (енгл. *Root Mean Squared Error - RMSE*) се често користи за евалуацију регресионих модела:

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^N (\hat{y}_i - y_i)^2} \quad (5.31)$$

при чему је N број узорака, y жељена односно права вредност, а \hat{y} предвиђена вредност. Наредна функција која се често користи за евалуацију регресионих модела је релативна квадратна грешка (енгл. *Relative Squared Error - RSE*):

$$RSE = \frac{\sum_{i=1}^N (\hat{y}_i - y_i)^2}{\sum_{i=1}^N (\bar{y} - y_i)^2} \quad (5.32)$$

при чему је N број узорака, y жељена односно права вредност, \hat{y} предвиђена вредност, а \bar{y} средња вредност жељених излаза модела. Степен корелације је метрика која мери снагу везе неке две променљиве. Може се користити као мера јачине везе између правих и предвиђених вредности:

$$r = \frac{\sum_{i=1}^N (\bar{y} - y_i)(\bar{\hat{y}} - \hat{y}_i)}{\sqrt{\sum_{i=1}^N (\bar{y} - y_i)^2 \sum_{i=1}^N (\bar{\hat{y}} - \hat{y}_i)^2}} \quad (5.33)$$

где је N број узорака, y жељена односно права вредност, \bar{y} аритметичка средина правих излазних вредности, \hat{y} предвиђена вредност, и $\bar{\hat{y}}$ аритметичка средина предвиђених излазних вредности.

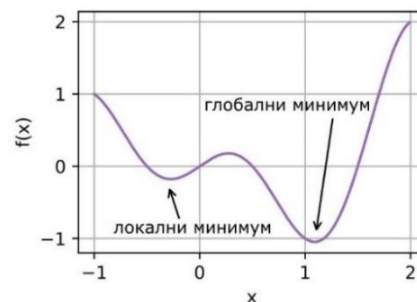
5.7 Алгоритми за оптимизацију

У овој секцији ће бити објашњени алгоритми за оптимизацију који се често користе за обуку неуронских мрежа. Скоро сви оптимизациони проблеми који се јављају у дубоком учењу су неконвексни. Без обзира на то, дизајн и анализа алгоритама за конвексне проблеме су се показали веома инструкивним. Обука компликованог модела може трајати сатима, данима или недељама. Перформансе алгорита за оптимизацију директно утичу на ефикасност обуке модела. Разумевање принципа функционисања различитих алгоритама, и улоге њихових хиперпараметара нам може омогућити да их подесимо на одговарајући начин тако да перформансе модела буду боље. Након што одаберемо функцију губитка, односно функцију циља, треба да одаберемо и алгоритама којим ћемо је минимизовати. По конвенцији већина алгоритама минимизује функцију, а уколико уместо тога треба максимизовати функцију циља, само треба променити знак функције. Иако нам оптимизација даје начин да минимизујемо функцију губитка, циљеви оптимизације и дубоког учења су суштински другачији. Грешка у току обуке и грешка приликом тестирања су често другачије, пошто је функција коју алгоритама минимизује заснована на скупу података за обуку. Циљ алгоритама за оптимизацију је да минимизује грешку у току обуке, а циљ дубоког учења је да минимизује генерализациону грешку. Да бисмо узели генерализацију у обзир, треба да обратимо пажњу и на преучавање, поред тога што минимизујемо функцију губитка.

Код дубоког учења функције циља су углавном компликоване и нема аналитичких решења, па је потребно користити нумеричке оптимизационе алгоритме. Постоје

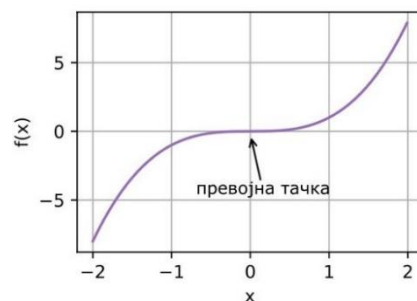
многи проблеми и изазови у оптимизацији код дубоког учења. Неки од проблема су локални минимуми и превојне тачке. За било коју функцију циља $f(x)$, ако је вредност у тачки x_m мања него вредности функције у било којој другој тачки у околини x_m , тада $f(x_m)$ може бити локални минимум. Ако је вредност $f(x_m)$ минимум функције циља преко целог њеног домена, тада је $f(x_m)$ глобални минимум. Функције циља могу имати мноштво локалних минимума.

У непосредној околини локалног минимума градијенти се приближавају нули, па зато када је нумеричко решење оптимизационог проблема у близини локалног оптимума, решење добијено у последњој итерацији ће можда минимизовати функцију циља локално, а не глобално. Неки облик шума може да избаци решење из области локалног минимума. На пример код стохастичког градијентног спуста са деловима скупа за обуку (енгл. *mini-batch stochastic gradient descend*) градијенти природно варијају од дела података до дела података, што може да омогући избегавање локалног минимума. Више детаља о овом алгоритму ће бити дато касније у овој секцији.



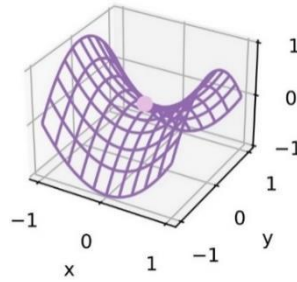
Слика 5.12 Пример глобалног и локалног минимума

Поред локалних минимума, превојне тачке могу бити разлог нестајања градијената. Превојна тачка је било која локација у којој сви градијенти функције нестају, али притом та локација није ни глобални ни локални минимум. На пример код функције $f(x) = x^3$, први и други извод нестају у тачки $x = 0$. Оптимизациони алгоритам може да се заглави у овој тачки, иако она не претставља минимум.



Слика 5.13 Превојна тачка функције $f(x) = x^3$

Превојне тачке у вишедимензионалним функцијама су још компликованије. Нпр. ако имамо функцију $f(x, y) = x^2 - y^2$ која има превојну тачку у $(0, 0)$, тада је ова тачка максимум по y , а минимум по x . Код неконвексних функција, вероватније је да се јави више превојних тачака него локалних минимума.

Слика 5.14 Превојна тачка функције $f(x, y) = x^2 - y^2$

5.7.1 Градијентни спуст и његове варијације

Нека је $f: \mathbb{R} \rightarrow \mathbb{R}$ континуално диференцијабилна реална функција, тада коришћењем Тејлоровог развоја важи:

$$f(x + \epsilon) = f(x) + \epsilon f'(x) + O(\epsilon^2) \quad (5.34)$$

Апроксимација првог реда је дата вредношћу функције $f(x)$ и првог извода у тачки x . За малу вредност ϵ померање у правцу негативном од правца градијента ће смањити f . Зарад једноставности одабраћемо фиксни корак $\eta > 0$ и $\epsilon = -\eta f'(x)$:

$$f(x - \eta f'(x)) = f(x) - \eta f'^2(x) + O(\eta^2 f'^2(x)) \quad (5.35)$$

Ако је извод различит од нуле, вредност функције ће се смањивати. Можемо одабрати довољно мало η тако да чланови вишег степена постану нерелевантни $f(x - \eta f'(x)) \lesssim f(x)$ и тада итерирамо по x :

$$x \leftarrow x - \eta f'(x) \quad (5.36)$$

Овим итерирањем вредност функције ће се смањивати. Дакле, најпре одаберемо иницијалну вредност x и константу η а затим итерирамо док апсолутна вредност градијента не постане довољно мала или док број итерација не достигне унапред задату вредност. Коефицијент η се назива стопом учења. Ако одаберемо малу вредност η биће потребан велики број итерација да се достигне решење. Ако је вредност η превелика, тада чланови вишег реда у Тејлоровом развоју нису занемариви, и не може се гарантовати да се итерирањем смањује вредност функције. Размотримо сада ситуацију где је $\mathbf{x} = [x_1, x_2, \dots, x_d]^T$, односно функција циља $f: \mathbb{R}^d \rightarrow \mathbb{R}$ мапира векторе у скаларну вредност. Одговарајући градијент је вишедимензионалан:

$$\nabla f(\mathbf{x}) = \left[\frac{\partial f(\mathbf{x})}{\partial x_1}, \frac{\partial f(\mathbf{x})}{\partial x_2}, \dots, \frac{\partial f(\mathbf{x})}{\partial x_d} \right]^T \quad (5.37)$$

Сваки парцијални извод $\frac{\partial f(\mathbf{x})}{\partial x_i}$ означава стопу промене у тачки \mathbf{x} у односу на улаз x_i . У вишедимензионалном случају важи да је:

$$f(\mathbf{x} + \epsilon) = f(\mathbf{x}) + \epsilon^T \nabla f(\mathbf{x}) + O(\|\epsilon\|^2) \quad (5.38)$$

Одабиром одговарајуће стопе учења η добијамо:

$$\mathbf{x} \leftarrow \mathbf{x} - \eta \nabla f(\mathbf{x}) \tag{5.39}$$

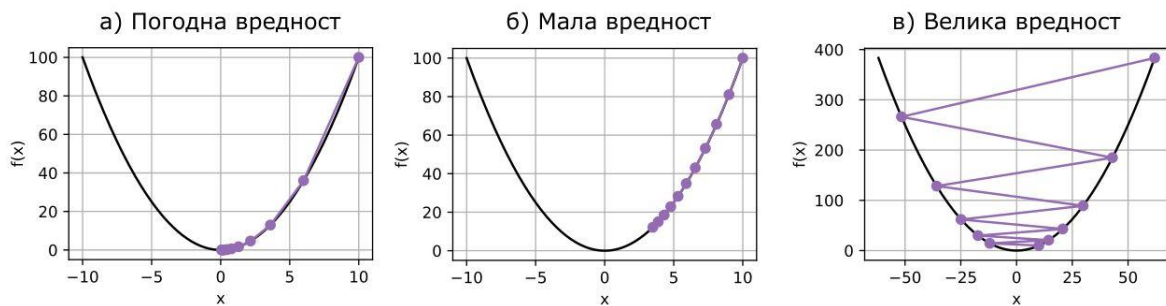
Одабир η може бити компликован. Сликвит приказ утицаја стопе учења је дат на слици 5.15. Постоје методе које разматрају и закривљеност поред вредности и градијента функције у посматраној тачки. Закривљеност захтева рачунање другог извода, па се ове методе зову методама другог реда. Ове методе су рачунски захтевније и ретко се користе у пракси. Код дубоког учења, функција циља је често средња вредност функција губитка за сваки узорак из скупа података за обуку. Ако је дат скуп за обуку са n узорака, претпоставимо да је $f_i(\mathbf{x})$ функција губитка која одговара i -том узорку, тада је функција циља:

$$f(\mathbf{x}) = \frac{1}{n} \sum_{i=1}^n f_i(\mathbf{x}) \tag{5.40}$$

а градијент је тада:

$$\nabla f(\mathbf{x}) = \frac{1}{n} \sum_{i=1}^n \nabla f_i(\mathbf{x}) \tag{5.41}$$

Утицај вредности стопе учења



Слика 5.15 Утицај вредности стопе учења на конвергенцију градијентног спуста

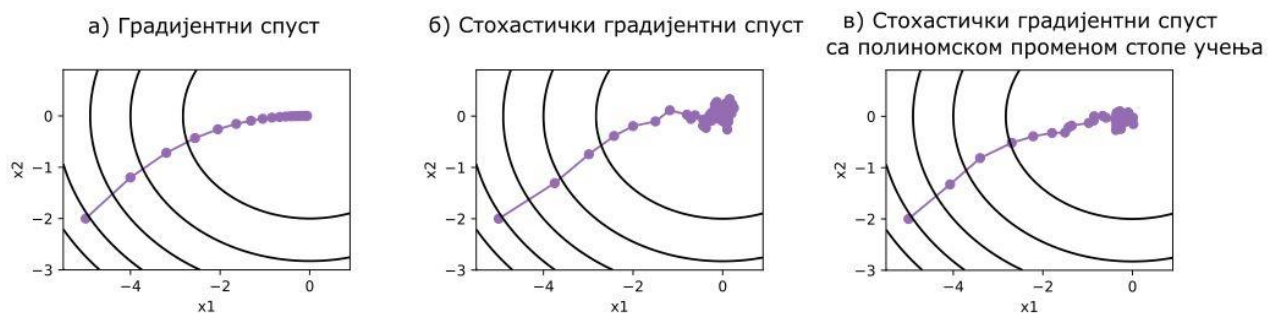
Када је скуп за обуку велики, градијентни спуст може да захтева пуно меморије. Стохастички градијентни спуст редукује меморијске захтеве тако што се у свакој итерацији, униформно одабере индекс $i \in \{1, \dots, n\}$, срачуна се градијент $\nabla f_i(\mathbf{x})$ и \mathbf{x} се ажурира по формули:

$$\mathbf{x} \leftarrow \mathbf{x} - \eta \nabla f_i(\mathbf{x}) \tag{5.42}$$

Трајекторија код стохастичког градијентног спуста може да има много више шума него трајекторија обичног градијентног спуста. Да би се ово разрешило треба прилагодити стопу учења заменом коефицијента η функцијом $\eta(t)$. Постоји више стратегија како стопа учења може да се мења у току времена, на пример: константно по деловима $\eta(t) = \eta_i$ за $t_i \leq t \leq t_{i+1}$, експоненцијално смањење $\eta(t) = \eta_0 e^{-\lambda t}$ и полиномско смањење $\eta(t) = \eta_0 (\beta t + 1)^{-\alpha}$. Када нема прогреса у току оптимизације, стопа учења треба да се смањи. Ово је стратегија која одговара промени стопе учења по деловима. Алтернативно, можемо смањивати стопу учења драстичније тј. експоненцијално. Експоненцијално смањење може да доведе до прераног заустављања алгорита, па се често користи полиномско смањење са $\alpha = 0.5$. На слици 5.16 је дат приказ обичног

градијентног спуста, стохастичког градијентног спуста и стохастичког градијентног спуста са полиномском стопом учења. Приказана је трајекторија, за произвољну функцију која као улаз прима вектор облика (x_1, x_2) . Види се да у трајекторији код стохастичког градијентног спуста постоји шум, који је смањен када се уведе полиномска промена стопе учења.

Стохастички градијентни спуст са деловима скупа података за обуку је комбинација два претходно наведена екстрема, градијентног спуста који користи цео скуп података у свакој итерацији, и стохастичког градијентног спуста који користи насумично одабран узорак у свакој итерацији. Главна мотивација за увођење ове модификације је рачунска ефикасност. Стохастички градијентни спуст је рачунски захтеван јер се параметри мреже ажурирају након процесирања сваког узорака, док се код градијентног спуста параметри ажурирају тек када је цео скуп података процесирани. Још једна мана стохастичког градијентног спуста су шумови који настају у градијентима. Процесирање узорак по узорак даје врло грубу процену стварног градијента који показује на правац у ком функција расте. Са друге стране градијентни спуст подразумева процесирање целог скупа података за обуку, што може да буде меморијски веома захтевно. Комбиновањем ова два приступа добијамо стабилнију конвергенцију и меморијски мање захтеван алгоритам. Када имамо део скупа података за обуку, градијент се усредњава по узорцима из тог дела података, што доприноси смањењу варијансе односно шума у процени градијента. Мана стохастичког градијента са деловима скупа података за обуку је та што је уведен додатни хиперпараметар. Корисник треба да одабере погодну величину дела скупа података који ће у једној итерацији проћи кроз неуронску мрежу.



Слика 5.16 Поређење трајекторије градијентног спуста и стохастичког градијентног спуста без и са полиномском променом стопе учења

5.7.2 Момент у алгоритмима за оптимизацију

Варијансу у процени градијента можемо додатно да смањимо ако нађемо неку врсту просека досадашњих градијената [53-59]. Нека је $\mathbf{g}_{t,t-1}$ градијент у кораку t који рачунамо у односу на параметре из корака $t-1$, тада можемо заменити рачунање градијента следећом формулом:

$$\mathbf{v}_t = \beta \mathbf{v}_{t-1} + \mathbf{g}_{t,t-1} \quad (5.43)$$

где је $\beta \in (0,1)$ параметар, а \mathbf{v} називамо моментом или импулсом. На овај начин акумулирамо претходне градијенте на сличан начин као што тешка лопта која се

котрља преко функције циља акумулира претходне силе, па отуда назив момент. Параметар β одређује утицај претходних градијената, на процену момента. Када је $\beta = 0$ методе са моментом се свODE на класичан градијентни спуст. Ако развијемо формулу (5.43) добићемо:

$$\mathbf{v}_t = \beta^2 \mathbf{v}_{t-2} + \beta \mathbf{g}_{t-1,t-2} + \mathbf{g}_{t,t-1} = \dots = \sum_{r=0}^{t-1} \beta^r \mathbf{g}_{t-r,t-r-1} \quad (5.44)$$

Нови градијент показује у правцу тежинске или пондерисане средње вредности претходних градијената. Методе које користе момент су се показале веома корисним у машинском учењу. Када имамо вишедимензионалне функције са улазним вектором (x_1, x_2, \dots, x_n) , промена у правцу x_i може бити много већа него промена у правцу x_j . Уколико користимо градијентни спуст без момента и одаберемо малу стопу учења имаћемо веома спору конвергенцију у правцу x_j , а са друге стране уколико одаберемо већу вредност стопе учења алгоритам ће дивергирати у правцу x_i . Овај проблем се решава увођењем момента. Са моментом, добијамо смањење промене у правцу x_i , јер усредњавамо осцилујуће градијенте који су у супротним правцима. У правцу x_j добијамо повећање промене, јер агрегирамо градијенте који су у истим правцима. Један од најпопуларнијих алгоритама који користе момент је Adam (назив добио од *adaptive momentum estimation*). Овај алгоритам поред првог момента користи и други момент градијента. Моменти су изражени следећим формулама:

$$\begin{aligned} \mathbf{v}_t &\leftarrow \beta_1 \mathbf{v}_{t-1} + (1 - \beta_1) \mathbf{g}_t, \\ \mathbf{s}_t &\leftarrow \beta_2 \mathbf{s}_{t-1} + (1 - \beta_2) \mathbf{g}_t^2 \end{aligned} \quad (5.45)$$

где је \mathbf{g}_t градијент у тренутном кораку алгоритма, а β_1 и β_2 су ненегативни параметри. Често се узимају вредности $\beta_1 = 0.9$ и $\beta_2 = 0.999$. За иницијалне вредности момената се узимају $\mathbf{v}_0 = 0$ и $\mathbf{s}_0 = 0$. Пошто постоји значајна тежња момената ка мањим вредностима, они се нормализују коришћењем формула:

$$\hat{\mathbf{v}}_t = \frac{\mathbf{v}_t}{1 - \beta_1^t} \quad \text{и} \quad \hat{\mathbf{s}}_t = \frac{\mathbf{s}_t}{1 - \beta_2^t} \quad (5.46)$$

и тада се нова вредност градијента изражава као:

$$\hat{\mathbf{g}}_t = \frac{\eta \hat{\mathbf{v}}_t}{\sqrt{\hat{\mathbf{s}}_t + \epsilon}} \quad (5.47)$$

где је η стопа учења, а ϵ врло мала вредност која спречава дељење нулом и омогућава нумеричку стабилност. Често се узима $\epsilon = 10^{-6}$. Ажурирање променљиве \mathbf{x}_t се изражава као:

$$\mathbf{x}_t \leftarrow \mathbf{x}_{t-1} - \hat{\mathbf{g}}_t \quad (5.48)$$

Исправљени Adam (енгл. *Rectified Adam*) је модификација Adam оптимизације која уноси додатни члан за исправљање варијансе у адаптивној стопи учења. Код Adam оптимизације се јавља висока варијанса у раној фази обуке модела, што може довести до лоше конвергенције. Алгоритам *Rectified Adam* је дат у листингу испод.

Алгоритам оптимизације - Rectified Adam

Улаз: $\eta_t, t = 1..T$ стопе учења, $\{\beta_1, \beta_2\}$ параметри за рачунање првог и другог момента, θ_0 иницијални параметри односно тежински коефицијенти, $f_t(\theta)$ стохастичка функција циља

Изназ: θ_t резултујући тежински коефицијенти

$v_0, s_0 \leftarrow 0$	Иницијализација момената
$p_\infty \leftarrow \frac{2}{1-\beta_2} - 1$	
while $t = \{1, \dots, T\}$ do	
$g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$	Срчунавање градијената
$s_t \leftarrow \beta_2 s_{t-1} + (1 - \beta_2) g_t^2$	Ажурирање другог момента
$v_t \leftarrow \beta_1 v_{t-1} + (1 - \beta_1) g_t$	Ажурирање првог момента
$\hat{v}_t = \frac{v_t}{1 - \beta_1^t}$	Корекција момента
$p_t \leftarrow p_\infty - 2t\beta_2^t / (1 - \beta_2^t)$	
if $p_t > 4$ then	
$l_t \leftarrow \sqrt{(1 - \beta_2^t) / s_t}$	Рачунање адаптивне стопе учења
$r_t \leftarrow \frac{\sqrt{(p_t - 4)(p_t - 2)p_\infty}}{\sqrt{(p_\infty - 4)(p_\infty - 2)p_t}}$	Рачунање члана за исправљање варијансе
$\theta_t \leftarrow \theta_{t-1} - l_t r_t \eta_t \hat{v}_t$	Ажурирање параметара
else	
$\theta_t \leftarrow \theta_{t-1} - \eta_t \hat{v}_t$	Ажурирање параметара
return θ_T	

Постоје и многи други алгоритми за оптимизацију, као што су на пример AdaGrad, AdaDelta, RMSProp, NAdam (*Adam with Nesterov momentum*), FTRL (*Follow The Regularized Leader*), Yogi и други. Иако је теорија о оптимизацији, напредовала последњих година, и даље постоје разни проблеми и изазови. Поред проблема као што су нестајући и експлодирајући градијенти, локални минимуми, превојне тачке и други равни региони, нетачни градијенти, итд. није потпуно јасно на који начин оптимизација утиче на генерализацију мреже, и не зна се унапред који оптимизациони алгоритам ће пронаћи тежинске коефицијенте такве да генерализација буде најбоља. Да би се поправила оптимизација, некада није најбоља стратегија побољшати директно оптимизациони алгоритам. Напредак у оптимизацији може доћи и креирањем модела који су лакши за оптимизовање. У пракси се показало да је важније одабрати модел који је погодан за оптимизацију, него користити софистициран оптимизациони алгоритам.

5.8 Неуронске мреже подржане физичким законима

Као универзални апроксиматори функција, неуронске мреже могу бити искоришћене за налажење решења парцијалних диференцијалних једначина. Захваљујући развоју аутоматске диференцијације и могућности да се добију изводи излаза неуронских мрежа по улазима, формиране су неуронске мреже подржане физичким законима. Обука ових типова неуронских мрежа је ограничена на тај начин да поштује симетрије, инваријантност или одржавање принципа који почивају на физичким законима исказаним у облику парцијалних диференцијалних једначина [60]. Неуронске мреже

подржане физичким законима се могу користити за директно решавање парцијалне диференцијалне једначине, са унапред задатим параметрима, али се такође могу користити и за инверзне проблеме где су параметри парцијалне диференцијалне једначине непознати, тако што уче на основу прикупљених података. Размотримо општу форму параметризоване временски зависне нелинеарне парцијалне диференцијалне једначине са граничним и почетним условима:

$$\begin{aligned} u_t + \mathcal{N}[u; \lambda] &= 0, x \in \Omega, t \in [0, T] \\ \text{Гранични услови: } \mathcal{B}(u, x) &= 0, x \in \partial\Omega, t \in [0, T] \\ \text{Почетни услови: } u(0, x) &= u_o(0, x), x \in \partial\Omega \end{aligned} \quad (5.49)$$

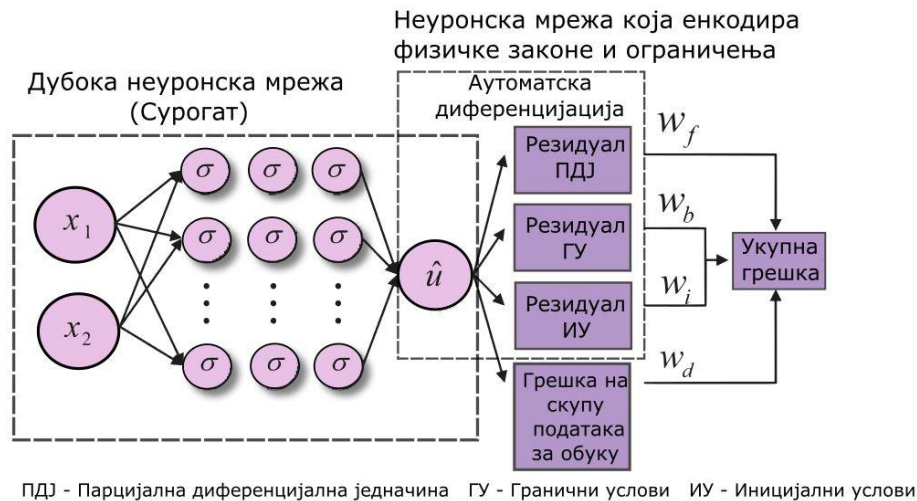
где је $\mathcal{N}[:, \lambda]$ нелинеарни оператор са параметрима λ , Ω је домен, а $\mathcal{B}(\cdot)$ је гранични услов. Нека је $NN_\theta(\cdot)$ неуронска мрежа са тежинским коефицијентима θ . Дефинишимо функцију која представља резидуал диференцијалне једначине:

$$f(t, x) := u_t + \mathcal{N}[u] \quad (5.50)$$

и наставимо са апроксимирањем функције $u(t, x)$ дубоком неуронском мрежом NN_θ . Функција $f(t, x)$ заправо представља физички закон енкодиран неуронском мрежом. Мрежа која енкодира физички закон, се формира применом ланчаног правила, и има исте параметре као мрежа NN_θ али другачије функције активације у зависности од операција диференцијалног оператора \mathcal{N} . Дељени параметри ових мрежа се добијају минимизовањем функције губитка:

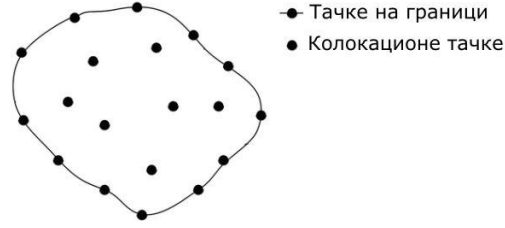
$$\begin{aligned} L(\theta) &= \omega_f L_{pde}(\theta; \mathcal{T}_f) + \omega_i L_{IC}(\theta; \mathcal{T}_i) + \omega_b L_{bc}(\theta; \mathcal{T}_b) + \omega_d L_{data}(\theta; \mathcal{T}_d) \\ L_{pde}(\theta; \mathcal{T}_f) &= \frac{1}{|\mathcal{T}_f|} \sum_{x \in \mathcal{T}_f} \|f(t, x)\|_2^2 \\ L_{IC}(\theta; \mathcal{T}_i) &= \frac{1}{|\mathcal{T}_i|} \sum_{x \in \mathcal{T}_i} \|\hat{u}(x) - u(x)\|_2^2 \\ L_{bc}(\theta; \mathcal{T}_b) &= \frac{1}{|\mathcal{T}_b|} \sum_{x \in \mathcal{T}_b} \|\mathcal{B}(\hat{u}, x)\|_2^2 \\ L_{data}(\theta; \mathcal{T}_{data}) &= \frac{1}{|\mathcal{T}_d|} \sum_{x \in \mathcal{T}_d} \|\hat{u}(x) - u(x)\|_2^2 \end{aligned} \quad (5.51)$$

где су $\omega_f, \omega_i, \omega_b, \omega_d$ тежински коефицијенти којима се множе одговарајуће компоненте функције губитка, $L_{pde}, L_{ic}, L_{bc}, L_{data}$ су редом функције губитка за водећу диференцијалну једначину, почетни услов, гранични услов и функција губитка на скупу података за обуку, $\mathcal{T}_f, \mathcal{T}_i, \mathcal{T}_b, \mathcal{T}_d$ су редом скупови тачака из домена водеће диференцијалне једначине, почетних, граничних услова и скупа података за обуку, \hat{u} је излаз мреже NN_θ . Крајњи циљ је пронаћи тежинске коефицијенте мреже NN_θ минимизовањем функције губитка. Шематски приказ неуронских мрежа подржаних физичким законима је дат на слици 5.17.



Слика 5.17 Шематски приказ неуронских мрежа подржаних физичким законима

Главна иновација код неуронских мрежа подржаних физичким законима је увођење резидуалне мреже која енкодира физички закон, узима излаз из дубоке неуронске мреже, коју називамо сурогатом, и рачуна резидуалну вредност. Основна формулација неуронских мрежа подржаних физичким законима не захтева никакве обележене податке, резултате других симулација или експерименталне податке. За неуронске мреже подржане физичким законима неопходна је процена резидуала дате диференцијалне једначине, као и почетних и граничних услова. Коришћење података, прикупљених из симулација или експеримената, за обучавање мреже је такође могуће, а и неопходно у неким случајевима, нарочито у инверзним проблемима. Подаци се обично користе у проблемима код којих гранични услови нису јасно дефинисани или фали једначина да би систем једначина био затворен. Након обуке, неуронска мрежа подржана физичким законом може бити коришћена као замена за традиционалне нумеричке солвере. Неуронске мреже подржане физичким законима представљају безмрежну методу, јер било која тачка у домену може бити узета као улаз без специфицирања повезаности између колокационих тачака. Једном обучена мрежа може бити коришћена за предвиђање вредности у чворовима мреже коначних елемената различитих резолуција без поновног обучавања, под условом да је генерализација неуронске мреже довољно добра. Код неуронских мрежа прорачуни са већим бројем чворова нису значајно рачунски захтевнији него прорачуни са мањим бројем чворова, као што је то случај код традиционалних нумеричких солвера. Код временски зависних проблема, време је представљено као било која друга променљива, па је могуће предвидети решење у временском тренутку од интереса, без решавања једначина у свим претходним временским тренуцима. Обука неуронских мрежа подржаних физичким законима подразумева генерисање или прикупљање тачака из домена од интереса. Тачака које су на граници домена, обично има значајно мање него тачака које су унутар домена, које називамо колокационим тачкама. Тачке на граници и колокационе тачке неког произвољног домена су приказане на слици 5.18.



Слика 5.18 Колокационе тачке и тачке на граници домена

Колокационе тачке нам служе да минимизујемо резидуал водеће диференцијалне једначине, док нам тачке на граници служе за минимизацију резидуала граничних услова. Уколико у формулацији губитка поставимо $\omega_f = \omega_b = 1$, утицај тачака на граници и колокационих тачака, на укупну функцију губитка неће бити једнак, због тога што је број колокационих тачака већи. У току обуке неуронске мреже, можемо посматрати утицај градијената компоненти функције губитка на ажурирање тежинских коефицијената мреже. Ако су градијенти који се односе на функцију губитка граничног услова веома мали у односу на градијенте који се односе на функцију губитка водеће једначине, као последица неједнаког броја тачака, то може довести до тога да гранични услов не буде испоштован. Без одговарајућих ограничења које дају почетни и гранични услови, парцијална диференцијална једначина може имати бесконачно много решења, па решење добијено неуронском мрежом, која није испоштвала граничне услове у току обуке, може бити потпуно нетачно. Да би се ово разрешило, развијане су разне стратегије за одређивање тежинских коефицијената $\omega_f, \omega_i, \omega_b, \omega_d$, којима се балансира утицај различитих компоненти на обуку мреже. Обележимо са L_r функцију губитка водеће једначине, а остале функције губитка обележимо са L_i за $i = 1, \dots, M$, где је M број компоненти које одговарају почетним, граничним условима, мерењима итд. У сваком кораку оптимизационог алгорита t , можемо срачунати $\hat{\omega}_i$:

$$\hat{\omega}_i = \frac{\max_{\theta} \{|\nabla_{\theta} L_r(\theta_t)|\}}{|\nabla_{\theta} L_i(\theta_t)|} \quad (5.52)$$

где $|\nabla_{\theta} L_i(\theta_t)|$ представља средњу вредност $|\nabla_{\theta} L_i(\theta_t)|$. Однос максималне вредности градијента резидуала водеће диференцијалне једначине $\max_{\theta} \{|\nabla_{\theta} L_r(\theta_t)|\}$ и средње вредности градијената i -те компоненте функције губитка, може пуно да варира. Због велике варијансе $\hat{\omega}_i$, тежински коефицијенти компонентни функције губитка ω_i се ажурирају преко формуле:

$$\omega_i = (1 - \alpha)\omega_i + \alpha\hat{\omega}_i \quad (5.53)$$

где је α параметар који се унапред задаје. Препоручена вредност за параметар α је 0.9. Уколико се за обуку мреже користи градијентни спуст, параметри односно тежински коефицијенти мреже се ажурирају према формули:

$$\theta_{t+1} = \theta_t - \eta \nabla_{\theta} L_r(\theta_t) - \eta \sum_{i=1}^M \omega_i \nabla_{\theta} L_i(\theta_t) \quad (5.54)$$

где η представља стопу учења. У имплементацији алгорита за ажурирање се често кориснику даје могућност да специфицира у којим корацима алгорита се ажурирају

кофицијенти ω_i , на пример у сваком петом или слично. Предност овог алгоритма је што није рачунски захтеван, нарочито када ажурирање ω_i није често. У пракси се показало да утицај параметра α није превелики, све док параметар узима вредности из разумног опсега као што је $\alpha \in [0.5, 0.9]$. Други алгоритам за ажурирање коефицијента ω_i је базиран на методи неуронских тангентних кернела (енгл. *Neural Tangent Kernel* скраћено НТК). Нека је неуронска мрежа дата функцијом $f(\cdot; \theta): \mathbb{R}^i \rightarrow \mathbb{R}^o$, где је i број улаза у мрежу, а o је број излаза из мреже, неуронски тангентни кернел је тада дефинисан као:

$$K_{k,l}(x, y; \theta) = \sum_{p=1}^{\mathbb{P}} \partial_{\theta_p} f_k(x; \theta) \partial_{\theta_p} f_l(y; \theta) \quad (5.55)$$

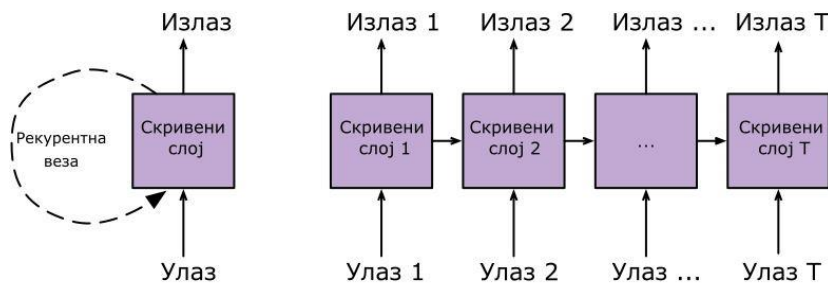
при чему је \mathbb{P} број тежинских коефицијената неуронске мреже. Ови кернели се користе за проучавање напретка дубоких мрежа у току оптимизације. Код неуронских мрежа подржаних физичким законима, рачунају се сопствене вредности кернела за сваку од компоненти укупне функције губитка, и те сопствене вредности се узимају за тежинске коефицијенте. Ова метода је рачунски захтевнија од претходне, али је теоријски поткованија.

Постоје различите библиотеке за рад са неуронским мрежама подржаних физичким законима, као што су *DeepXDE*, *SimNet*, *NeuralPDE* и *SciANN*. *DeepXDE* нуди олакшано коришћене неуронских мрежа подржаних физичким законима за решавање обичних и парцијалних диференцијалних једначина, као и за решавање инверзних проблема. Поред вишеслојног перцептрона, нуди и мреже са вишескалним Фуријеовим карактеристикама, резидуалне мреже итд. *DeepXDE* даје кориснику могућност да дефинише комплексне геометријске домене комбиновањем основних као што су интервал, троугао, правоугаоник, полигон, диск, сфера, хиперкоцка и хиперсфера. У овој библиотеци су доступни различити типови граничних услова (Дирихлеов, Нојманов, Робин, периодични и општи гранични услов за дефинисање на произвољном домену), различити алгоритми за оптимизацију, различите методе узорковања тачака из задатог домена итд. *SimNet* је развијен од стране компаније *Nvidia* и оптимизован је за њихове графичке процесоре. *NeuralPDE* је писан у програмском језику *Julia*, пружа могућност дефинисања нове функције губитка која комбинује учење на основу једначина и на основу података. Такође, ова библиотека пружа могућност рада са парцијалним интегро-диференцијалним једначинама и стохастичким једначинама. *SciANN* је библиотека писана у *Python*-у, која се може користити за решавање обичних и парцијалних диференцијалних једначина, за инверзне проблеме, као и за фитовање кривих коришћењем вишеслојног перцептрона. *SciANN* се ослања на библиотеке *Keras* и *Tensorflow* које се често користе за рад са неуронских мрежама. Приликом израде ове дисертације коришћене су библиотеке *SciANN* и *DeepXDE*, јер су се показале једноставним за употребу.

5.9 Рекурентне неуронске мреже

Многи проблеми учења из података захтевају рад са секвенцијалним подацима. Титловање слика, синтеза говора, генерисање музике итд. захтевају да резултат предикције буде низ података. Анализа временских серија, анализа видео и аудио

записа, захтева да улаз у модел буде секвенцијални низ података. Некада ови захтеви, да улаз у неуронску мрежу и излаз из ње буде секвенцијалан, иду заједно, као што је то случај са превођењем делова текста из једног језика у други, вођење дијалога, контролисање робота итд. Рекурентне неуронске мреже су тип неуронских мрежа које уче из секвенцијалних података, користећи рекурентне везе [61]. Ове мреже се могу посматрати као мреже које се одмотавају преко секвенце, при чему се на сваки члан секвенце примењују исте операције. Док стандардне везе служе за пропацију сигнала кроз мрежу за један члан секвенце, односно један тренутак, рекурентне везе пропацирају сигнал који се односи на суседне чланове секвенце односно на суседне временске тренутке. Када развијемо рекурентну мрежу, можемо да је посматрамо као вишеслојни перцептрон, где се свака веза, било рекурентна или стандардна веза, односи на све чланове секвенце, односно на све временске тренутке. Овај тип неуронских мрежа се најчешће користи за процесирање природног језика.



Слика 5.19 Шематски приказ развијене рекурентне мреже

Сваки од улазних атрибута, неуронских мрежа за анализу секвенцијалних података, је уређена листа података поређаних по кораку секвенце. Неки скупови података се састоје од једне веома дуге секвенце, као што су на пример читавања са сензора, док други скупови података представљају колекцију секвенци, као што су то нпр. евиденције боравка пацијената у болницама. Код обичне несеквенцијалне анализе података, сваки индивидуални улаз се посматра независно од целе дистрибуције података. Код анализе секвенци, сваки узорак секвенце се и даље посматра независно, али унутар једне секвенце се не може претпоставити да не постоји међусобна зависност једног секвенцијалног корака од другог. На пример, која реч ће се појавити у неком писму, увелико зависи од тога које речи су претходно коришћене. Који лек пацијент треба да добије десетог дана у болници, зависи од тога шта се дешавало претходних девет дана. Некад треба предвидети податак у на основу секвенцијално структурираних података (нпр. класификација сентимента на основу критике филма), некада треба предвидети секвенцу (y_1, y_2, \dots, y_r) на основу фиксног улаза (титловање слике), а некада треба предвидети секвенцу на основу дате улазне секвенце (машинско превођење, или титловање видео клипова).

Нека вредност x_t у тренутку t зависи само од $n - 1$ претходних вредности. Уколико желимо да моделирамо утицај већег броја претходних догађаја, треба да повећамо n . Међутим, број параметара модела се тада експоненцијално повећава, зато уместо да моделирамо $P(x_t | x_{t-1}, \dots, x_{t-n+1})$, корисимо латентни модел:

$$P(x_t | x_{t-1}, \dots, x_1) \approx P(x_t | h_{t-1}) \quad (5.56)$$

где је h_{t-1} скривено стање које чува информацију секвенце до корака $t - 1$. У општем случају скривено стање у било ком тренутку t може да буде срачунато на основу тренутног улаза x_t и претходног скривеног стања h_{t-1} :

$$h_t = f(x_t, h_{t-1}) \quad (5.57)$$

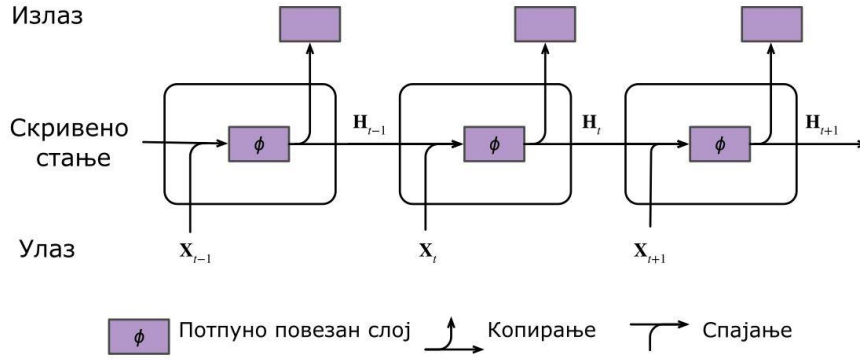
За довољно добру функцију f латентни модел није апроксимација. У крајњем случају h_t може да чува све податке посматране до датог тренутка. Технички, скривена стања су улази за тренутни временски корак, а могу бити срачуната само процесирањем улазних података из претходних временских корака. Рекурентне неуронске мреже користе скривена стања. Нека имамо део улаза $\mathbf{X}_t \in \mathbb{R}^{n \times d}$ у тренутку t , при чему је n величина улаза, а d број улазних атрибута. За групу од n секвенцијалних узорака, сваки ред \mathbf{X}_t одговара једном узорку у тренутку t из секвенце. Нека је $\mathbf{H}_t \in \mathbb{R}^{n \times d}$ излаз скривеног слоја у тренутку t . За разлику од вишеслојног перцептрона, овде чувамо излаз из претходног временског корака \mathbf{H}_{t-1} и уводимо нов тежински коефицијент $\mathbf{W}_{hh} \in \mathbb{R}^{h \times h}$ да опишемо како излаз скривеног слоја из претходног корака утиче на тренутни. Прецизније, рачунање излаза тренутног скривеног слоја је одређено улазом за тренутни временски корак заједно са излазом скривеног слоја за претходни временски корак:

$$\mathbf{H}_t = \phi(\mathbf{X}_t \mathbf{W}_{xh} + \mathbf{H}_{t-1} \mathbf{W}_{hh} + \mathbf{b}_h) \quad (5.58)$$

где су \mathbf{W}_{xh} тежински коефицијенти, \mathbf{b}_h слободни коефицијент и h је број скривених јединица у скривеном слоју. Из односа између излаза скривених слојева \mathbf{H}_t и \mathbf{H}_{t-1} за суседне временске тренутке, знамо да су ове променљиве научиле и сачувале секвенцијалне информације, у виду стања или меморије неуронске мреже. Зато се овакав излаз скривеног слоја назива скривеним стањем. Једначина је рекурентна јер иста дефиниција важи за сваки корак. Слојеви који врше рекурентне операције се називају рекурентним слојевима. За временски корак t , излаз последњег слоја је сличан као излаз из вишеслојног перцептрона:

$$\mathbf{O}_t = \mathbf{H}_t \mathbf{W}_{hq} + \mathbf{b}_q \quad (5.59)$$

при чему је q број излазних атрибута. Параметри рекурентних мрежа укључују тежинске коефицијенте $\mathbf{W}_{xh} \in \mathbb{R}^{d \times h}$, $\mathbf{W}_{hh} \in \mathbb{R}^{h \times h}$ и слободне коефицијенте $\mathbf{b}_h \in \mathbb{R}^{1 \times h}$ скривених слојева, заједно са тежинским коефицијентима $\mathbf{W}_{hq} \in \mathbb{R}^{h \times q}$ и слободним коефицијентима $\mathbf{b}_q \in \mathbb{R}^{1 \times q}$ излазног слоја. Треба напоменути да и у другим временским корацима, рекурентне мреже користе ове исте параметре, што значи да број параметара рекурентне мреже не расте са бројем временских корака који се посматрају. Слика 5.20 илуструје рачунску логику рекурентне мреже за три суседна временска корака. У било ком тренутку t , срачунавање скривеног стања може бити третирано као надовезивање улаза \mathbf{X}_t за тренутни корак и скривеног стања за претходни корак \mathbf{H}_{t-1} , а затим пролазак надовезаног резултата кроз потпуно повезан слој са активационом функцијом ϕ . Излаз овог потпуно повезаног слоја је скривено стање за тренутни тренутак \mathbf{H}_t . У овом случају параметри модела су надовезани тежински коефицијенти $\mathbf{W}_{xh} \in \mathbb{R}^{d \times h}$, $\mathbf{W}_{hh} \in \mathbb{R}^{h \times h}$ и слободни коефицијенти $\mathbf{b}_h \in \mathbb{R}^{1 \times h}$. Скривено стање тренутног временског корака \mathbf{H}_t , ће учествовати у срачунавању скривеног стања за наредни корак \mathbf{H}_{t+1} . Скривено стање ће такође бити коришћено за рачунање излаза из излазног слоја \mathbf{O}_t за тренутни корак t .



Слика 5.20 Развијена рекурентна мрежа са приказаним скривеним стањима [44]

5.9.1 Одсецање градијената

Код дубоких неуронских мрежа, многи слојеви раздвајају улазни и излазни слој. Дужина секвенце уводи нову димензију дубине. Поред тога што секвенца пролази кроз слојеве мреже у правцу излаз-улаз, улазни тензор за први временски корак пролази и кроз слојеве мреже дуж T корака секвенце. У свакој итерацији пропагирамо градијенте уназад кроз време, што значи да имамо $O(T)$ матричних множења, чиме се може добити нумеричка нестабилност у виду експлодирајућих или нестајућих градијената у зависности од особина матрица тежинских коефицијената. Рад са нестајућим и експлодирајућим градијентима је фундаментални проблем приликом дизајнирања рекурентних неуронских мрежа и инспирисао је многе иновације у архитектурама модерних неуронских мрежа. Специјализоване архитектуре које су отпорније на проблеме нестајућих градијената ће бити касније представљене. Међутим, чак и те специјалне архитектуре нису имуне на проблем експлодирајућих градијената. Једно елегантно решење је одсецање градијената. Када треба оптимизовати неку функцију губитка f , градијентним спустом, ажурирају се параметри од интереса x у правцу супротном од градијента g , са стопом учења $\eta > 0$ формулом $x \leftarrow x - \eta g$. Можемо претпоставити да је функција губитка довољно глатка. Формално, кажемо да је функција Липшиц континуална са константном L у значењу да за свако x и y имамо:

$$|f(x) - f(y)| \leq L \|x - y\| \quad (5.60)$$

Када ажурирамо вектор параметара, одузимањем ηg , промена вредности функције губитка зависи од стопе учења, а норма градијената и L је :

$$|f(x) - f(x - \eta g)| \leq L \eta \|g\| \quad (5.61)$$

Другим речима функција губитка се не може променити за вредност већу од $L \eta \|g\|$. Ограничавање горње границе може имати и предности и мане. Са једне стране, смањује се брзина којим се редукује вредност функције губитка, а са друге стране ограничена је и величина грешке која може бити направљена у једном кораку. Када се каже да градијенти експлодирају мисли се да норма $\|g\|$ постаје веома велика. У најгорим случајевима, може бити направљена толика грешка да се изгуби сав прогрес у хиљадама итерација обуке неурона. Када су градијенти тако велики, обука може да дивергира. Често се дешава и да обука конвергира, али је нестабилна и подложна наглим скоковима у функцији губитка. Један од начина да се смањи вредност $L \eta \|g\|$ је смањење стопе учења η . Тај приступ је проблематичан ако се високи градијенти ретко добијају, јер тада успоравамо прогрес у свим корацима, само да бисмо избегли ретку појаву експлодирања градијената. Популарна алтернатива је усвајање хеуристике одсецања градијената, пројектовањем градијената g на лопту са радијусом θ :

$$g \leftarrow \min(1, \frac{\theta}{\|g\|}) g \quad (5.62)$$

Ово омогућава да норма градијента никада не прелази θ и да је ажирирани градијент увек у правцу оригиналног вектора \mathbf{g} .

5.9.2 Пропагација уназад кроз време

Пропагација уназад код рекурентних мрежа се назива пропагација уназад кроз време. Ова процедура захтева развијање или експанзију рачунског графа рекурентне мреже на секвенцу корака, и примену ланчаног правила извода, пропагацију градијената кроз развијену мрежу. Градијенти по сваком параметру морају бити сумирани за све параметре који се јављају у развијеној мрежи. Компликације се јављају јер дужина секвенци може бити велика. Улази из првог корака могу пролазити кроз хиљаде матричних производа пре него што стигну до излаза мреже, и још хиљаде матричних производа је потребно за рачунање градијената. Нека су h_t, x_t, o_t скривено стање, улаз и излаз из мреже у тренутку t . Улаз и скривено стање могу бити надовезани једно на друго, пре него што буду помножени тежинским коефицијентима у скривеном слоју. Са w_h и w_o обележићемо тежинске коефицијенте скривеног и излазног слоја, респективно. Скривено стање и излази су тада:

$$h_t = f(x_t, h_{t-1}, w_h) \quad (5.63)$$

$$o_t = g(h_t, w_o) \quad (5.64)$$

где су f и g трансформације скривеног и излазног слоја, респективно. Дакле, имамо ланац вредности $\{\dots, (x_{t-1}, h_{t-1}, o_{t-1}), (x_t, h_t, o_t), \dots\}$ које зависе једне од других преко рекурентног прорачуна. Пропагација унапред се састоји од тога да прођемо кроз триплете (x_t, h_t, o_t) корак по корак. Разлика између излаза o_t и жељене вредности y_t се евалуира функцијом губитка за T временских корака:

$$L(x_1, \dots, x_T, y_1, \dots, y_T, w_h, w_o) = \frac{1}{T} \sum_{t=0}^T l(y_t, o_t). \quad (5.65)$$

Пропагација уназад је нешто компликованија, посебно када се рачунају градијенти функције губитка L по w_h . По ланчаном правилу важи:

$$\frac{\partial L}{\partial w_h} = \frac{1}{T} \sum_{t=0}^T \frac{\partial l(y_t, o_t)}{\partial w_h} = \frac{1}{T} \sum_{t=0}^T \frac{\partial l(y_t, o_t)}{\partial o_t} \frac{\partial g(h_t, w_o)}{\partial h_t} \frac{\partial h_t}{\partial w_h}. \quad (5.66)$$

Први и други фактор производа (5.66) су лаки за израчунавање, али код трећег фактора $\frac{\partial h_t}{\partial w_h}$ настају компликације, пошто је потребно рекурентно срачунати ефекат параметра w_h на h_t . Скривено стање h_t зависи од h_{t-1} и од w_h , при чему и h_{t-1} такође зависи од w_h . Коришћењем ланчаног правила за израчунавање извода h_t по w_h добија се:

$$\frac{\partial h_t}{\partial w_h} = \frac{\partial f(x_t, h_{t-1}, w_h)}{\partial w_h} + \frac{\partial f(x_t, h_{t-1}, w_h)}{\partial h_{t-1}} \frac{\partial h_{t-1}}{\partial w_h}. \quad (5.67)$$

Претпоставимо да имамо секвенце $\{a_t\}, \{b_t\}, \{c_t\}$ које задовољавају $a_0 = 0$ и $a_t = b_t + c_t a_{t-1}$ за $t = 1, 2, \dots$ тада се може лако показати да је:

$$a_t = b_t + \sum_{i=1}^{t-1} \left(\prod_{j=t+1}^i c_j \right) b_t. \quad (5.68)$$

Нека су:

$$a_t = \frac{\partial h_t}{\partial w_h}, b_t = \frac{\partial f(x_t, h_{t-1}, w_h)}{\partial w_h}, c_t = \frac{\partial f(x_t, h_{t-1}, w_h)}{\partial h_{t-1}} \quad (5.69)$$

Тада градијент у једначини (5.66) задовољава $a_t = b_t + c_t a_{t-1}$. Дакле по једначини (5.68) можемо заменити рекурентни прорачун у (5.66) са:

$$\frac{\partial h_t}{\partial w_h} = \frac{\partial f(x_t, h_{t-1}, w_h)}{\partial w_h} + \sum_{i=1}^{t-1} \left(\prod_{j=t+1}^i \frac{\partial f(x_j, h_{j-1}, w_h)}{\partial h_{j-1}} \right) \frac{\partial f(x_i, h_{i-1}, w_h)}{\partial w_h}. \quad (5.70)$$

Овај ланац може постати веома дугачак када је број временских корака велики, и његово срачунавање може бити веома споро. Осим тога, градијенти могу постати огромни, пошто мале измене у иницијалним условима могу доста да промене коначну вредност. Постоји пар стратегија којима се овај проблем може решити. Једно од решења је скраћивање суме у једначини (5.70) након неког броја корака, што доводи до апроксимације градијената. Мана овог приступа је фокусирање утицаја скоријих догађаја на предикцију, и занемаривање дугорочних узрочно-последичних веза. Још једна од опција је замена $\frac{\partial h_t}{\partial w_h}$ насумичном променљивом која има сличну очекивану вредност али скраћује секвенцу. Ово се постиже коришћењем секвенце ξ_t са унапред дефинисаним $0 \leq \pi_t \leq 1$ где су $P(\xi_t = 0) = 1 - \pi_t$ и $P(\xi_t = \pi_t^{-1}) = \pi_t$ одакле је $E|\xi_t| = 1$. Ово омогућава замену градијента $\frac{\partial h_t}{\partial w_h}$ у једначини (5.67) са:

$$z_t = \frac{\partial f(x_t, h_{t-1}, w_h)}{\partial w_h} + \xi_t \frac{\partial f(x_j, h_{j-1}, w_h)}{\partial h_{j-1}} \frac{\partial h_{t-1}}{\partial w_h}. \quad (5.71)$$

Из дефиниције ξ_t следи да је $E|z_t| = \frac{\partial h_t}{\partial w_h}$. Када је $\xi_t = 0$, рекурентни прорачун се завршава у том кораку t . Ово води до пондерисане суме секвенци различитих дужина, где су дуге секвенце ретке али и сразмерно већих тежинских коефицијената. Овај приступ у пракси ипак не функционише много боље од обичног регуларног одсецања суме. Повећана варијанса се коси са чињеницом да је градијент тачнији са више временских корака. Након представљања опште идеје, посматрајмо процес пропагације уназад кроз време детаљније. Зарад једноставности узећемо рекурентну мрежу без слободних коефицијената, у чијим се скривеним слојевима као активациона функција користи функција идентитета ($\phi(x) = x$). За корак t , улаз у мрежу је $x_t \in \mathbb{R}^d$ а излаз је y_t . Скривено стање $\mathbf{h}_t \in \mathbb{R}^h$ и излаз $\mathbf{o}_t \in \mathbb{R}^q$ се рачунају као:

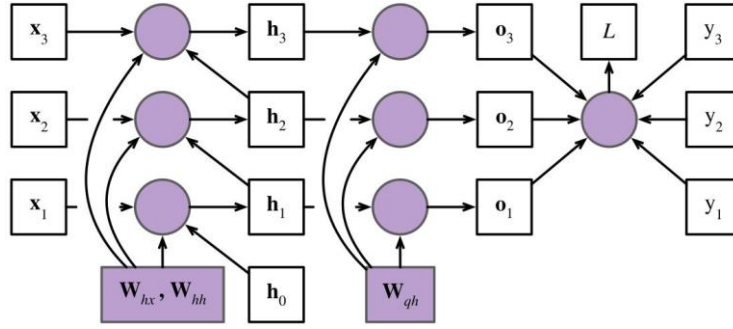
$$\mathbf{h}_t = \mathbf{W}_{hx} \mathbf{x}_t + \mathbf{W}_{hh} \mathbf{h}_{t-1} \quad (5.72)$$

$$\mathbf{o}_t = \mathbf{W}_{qh} \mathbf{h}_t \quad (5.73)$$

где су $\mathbf{W}_{hx} \in \mathbb{R}^{h \times d}$, $\mathbf{W}_{hh} \in \mathbb{R}^{h \times h}$, $\mathbf{W}_{qh} \in \mathbb{R}^{q \times h}$ тежински коефицијенти односно параметри мреже. Обележимо функцију губитка у тренутку t са $l(\mathbf{o}_t, y_t)$. Функција губитка за T временских корака од почетка секвенце је тада:

$$L = \frac{1}{T} \sum_{t=1}^T l(\mathbf{o}_t, y_t) \quad (5.74)$$

Да бисмо визуализовали зависност променљивих и параметара, на слици 8.21 је приказан граф прорачуна. На пример, рачунање скривеног стања \mathbf{h}_3 у кораку 3, зависи од параметара \mathbf{W}_{hx} , \mathbf{W}_{hh} , \mathbf{W}_{qh} , скривеног стања \mathbf{h}_2 , и улаза у тренутни корак \mathbf{x}_3 . Обука мреже захтева прорачун градијената $\frac{\partial L}{\partial w_{hx}}$, $\frac{\partial L}{\partial w_{hh}}$ и $\frac{\partial L}{\partial w_{qh}}$. На слици 8.21, можемо посматрати правце супротне стрелицама да бисмо израчунали потребне градијенте. Да бисмо флексибилније изразили множење матрица, вектора и скалара различитих димензија у ланчаним изводима, користићемо оператор Π .



Слика 5.21 Граф прорачуна са зависностима рекурентне мреже за три временска корака [44]

Налажење извода функције губитка по излазу мреже у неком кораку t врши се по формули:

$$\frac{\partial L}{\partial \mathbf{o}_t} = \frac{\partial l(\mathbf{o}_t, \mathbf{y}_t)}{\partial \mathbf{o}_t} \in \mathbb{R}^q \quad (5.75)$$

Градијент функције губитка по параметру \mathbf{W}_{qh} се рачуна помоћу ланчаног правила:

$$\frac{\partial L}{\partial \mathbf{W}_{qh}} = \sum_{t=1}^T \prod \left(\frac{\partial L}{\partial \mathbf{o}_t}, \frac{\partial \mathbf{o}_t}{\partial \mathbf{W}_{qh}} \right) = \sum_{t=1}^T \frac{\partial L}{\partial \mathbf{o}_t} \mathbf{h}_t^T \quad (5.76)$$

У последњем временском кораку T , функција губитка L зависи од скривеног стања \mathbf{h}_T само преко \mathbf{o}_T и тада можемо наћи градијент $\partial L / \partial \mathbf{h}_T$ коришћењем ланчаног правила извода:

$$\frac{\partial L}{\partial \mathbf{h}_T} = \prod \left(\frac{\partial L}{\partial \mathbf{o}_T}, \frac{\partial \mathbf{o}_T}{\partial \mathbf{h}_T} \right) = \mathbf{W}_{qh}^T \frac{\partial L}{\partial \mathbf{o}_T} \quad (5.77)$$

За тренутке $t < T$ где функција губитка L зависи од \mathbf{h}_t преко \mathbf{h}_{t+1} и \mathbf{o}_t , градијент се рачуна рекурентно:

$$\frac{\partial L}{\partial \mathbf{h}_t} = \prod \left(\frac{\partial L}{\partial \mathbf{h}_{t+1}}, \frac{\partial \mathbf{h}_{t+1}}{\partial \mathbf{h}_t} \right) + \prod \left(\frac{\partial L}{\partial \mathbf{o}_t}, \frac{\partial \mathbf{o}_t}{\partial \mathbf{h}_t} \right) = \mathbf{W}_{hh}^T \frac{\partial L}{\partial \mathbf{h}_{t+1}} + \mathbf{W}_{qh}^T \frac{\partial L}{\partial \mathbf{o}_t} \quad (5.78)$$

У општем случају важи:

$$\frac{\partial L}{\partial \mathbf{h}_t} = \sum_{i=1}^T (\mathbf{W}_{hh}^T)^{T-i} \mathbf{W}_{qh}^T \frac{\partial L}{\partial \mathbf{o}_{T+i-t}} \quad (5.79)$$

Ова једнакост у себи садржи високе степене \mathbf{W}_{hh}^T , што може довести до проблема са нестајућим и експлодирајућим градијентима. Ово се решава, као што је раније напоменуто, скраћивањем броја временских корака на рачунски погодну величину. Функција губитка L зависи и од параметара \mathbf{W}_{hx} , \mathbf{W}_{hh} у скривеним слојевима преко скривених стања $\mathbf{h}_1, \dots, \mathbf{h}_T$. За рачунање градијената користе се формуле:

$$\frac{\partial L}{\partial \mathbf{W}_{hx}} = \sum_{t=1}^T \prod \left(\frac{\partial L}{\partial \mathbf{h}_t}, \frac{\partial \mathbf{h}_t}{\partial \mathbf{W}_{hx}} \right) = \sum_{t=1}^T \frac{\partial L}{\partial \mathbf{h}_t} \mathbf{x}_t^T \quad (5.80)$$

$$\frac{\partial L}{\partial \mathbf{W}_{hh}} = \sum_{t=1}^T \prod \left(\frac{\partial L}{\partial \mathbf{h}_t}, \frac{\partial \mathbf{h}_t}{\partial \mathbf{W}_{hh}} \right) = \sum_{t=1}^T \frac{\partial L}{\partial \mathbf{h}_t} \mathbf{h}_{t-1}^T \quad (5.81)$$

где се $\partial L / \partial \mathbf{h}_t$ рекурентно рачуна као у једначинама (5.75) и (5.76). У имплементацији пропагације уназад кроз време многи међурезултати се чувају, да би се избегли дупли прорачуни, чиме се између осталог штеди на рачунском времену. На пример, сачувани резултат $\partial L / \partial \mathbf{h}_t$ се може искористити за срачунавање $\frac{\partial L}{\partial \mathbf{W}_{hx}}$ и $\frac{\partial L}{\partial \mathbf{W}_{hh}}$.

5.10 Унапређене верзије рекурентних неуронских мрежа

Нумеричка нестабилност приликом обуке обичних рекурентних неуронских мрежа је довела до мотивације за развој других софистициранијих архитектура [62]. Неке од тих мрежа су затворена рекурентна јединица (енгл. *Gated Recurrent Units* или скраћено *GRU*) и јединица са дуготрајном-краткотрајном меморијом (енгл. *Long Short-Term Memory* или скраћено *LSTM*). Поред рекурентних мрежа код којих се рекурентне операције одвијају у једном смеру, постоје и бидирекционе рекурентне мреже, код којих се рекурентне операције одвијају у оба смера – напред и уназад.

5.10.1 Затворене рекурентне јединице

Дискутовали смо о проблемима нестајућих и експлодирајућих градијената који се јављају услед великог броја множења матрица код рекурентних мрежа. У пракси можемо имати разне проблеме у којима се јављају аномалије са градијентима. На пример, можемо наићи на такав тип проблема где је први узорак веома значајан за предвиђање. У том случају, пошто то прво опажање утиче на наредна опажања, можемо добити велики градијент који истиче његову значајност, што би било добро избећи. Рецимо да прво опажање представља број који контролише тачност суме, и да је крајњи циљ да одредимо да ли је резултат добар. У овом случају, то прво опажање је веома значајно и било би добро да можемо да га сачувамо у неку меморијску јединицу, чиме избегавамо добијање великог градијента. Такође, можемо наићи на ситуације где неки узорци не представљају специјално релевантно опажање, па би у тим случајевима било добро да можемо такве узорке да прескочимо. На крају, можемо наилазити на ситуације где постоји логички прекид у деловима секвенце, као што би то, на пример, били прелази између поглавља књиге. У овим случајевима би било добро да имамо механизам за ресетовање интерног стања. Многе методе су предложене као решења за овакве ситуације. Једна од ранијих метода је јединица са дуготрајном-краткотрајном меморијом. Затворена рекурентна јединица је новија метода која даје сличне перформансе али је мање рачунски захтевна [63]. Због једноставности кренућемо од ње. Главна разлика између затворене и обичне рекурентне јединице, је што затворена има механизме за одлучивање када скривено стање треба бити ажурирано, а када ресетовано.

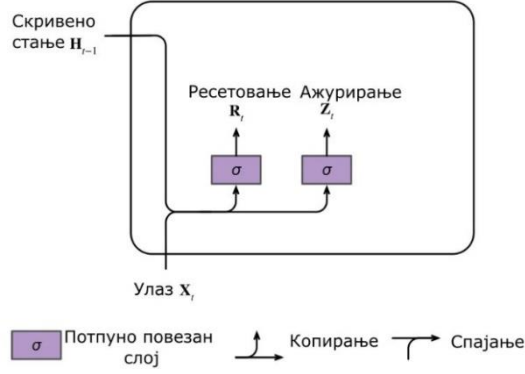
5.10.1.1 Механизми за ресетовање и ажурирање

Механизам за ресетовање омогућава да рекурентна јединица научи који удео старог стања треба да упамти, док механизам за ажурирање омогућава контролу удела сличности новог стања са старим стањем. На слици 5.22 су приказани механизми за ресетовање и ажурирање. На основу тренутног улаза и скривеног стања из претходног временског корака, рачунају се излази механизма, односно кола, која су дата као два потпуно повезана слоја са сигмоидалном активационом функцијом. Математички, за дати временски корак t , за део улаза $\mathbf{X}_t \in \mathbb{R}^{n \times d}$ (n је број узорака, d је број улаза) и скривено стање $\mathbf{H}_{t-1} \in \mathbb{R}^{n \times h}$ (h је број скривених јединица) из претходног временског корака, излази механизма за ресетовање \mathbf{R}_t и за ажурирање \mathbf{Z}_t су:

$$\mathbf{R}_t = \sigma(\mathbf{X}_t \mathbf{W}_{xr} + \mathbf{H}_{t-1} \mathbf{W}_{hr} + \mathbf{b}_r) \quad (5.82)$$

$$\mathbf{Z}_t = \sigma(\mathbf{X}_t \mathbf{W}_{xz} + \mathbf{H}_{t-1} \mathbf{W}_{hz} + \mathbf{b}_z) \quad (5.83)$$

где су $\mathbf{W}_{xr}, \mathbf{W}_{xz} \in \mathbb{R}^{d \times h}$ и $\mathbf{W}_{hr}, \mathbf{W}_{hz} \in \mathbb{R}^{h \times h}$ тежински коефицијенти, а $\mathbf{b}_r, \mathbf{b}_z$ слободни коефицијенти. Сигмоидална функција се користи да би опсег улазних вредности био трансформисан на интервал $(0,1)$.



Слика 5.22 Механизми за ресетовање и ажурирање затворене рекурентне јединице [44]

5.10.1.2 Кандидат за скривено стање

Кандидат за скривено стање $\tilde{\mathbf{H}}_t$ у тренутку t се добија као:

$$\tilde{\mathbf{H}}_t = \tanh(\mathbf{X}_t \mathbf{W}_{xh} + (\mathbf{R}_t \odot \mathbf{H}_{t-1}) \mathbf{W}_{hh} + \mathbf{b}_h) \quad (5.84)$$

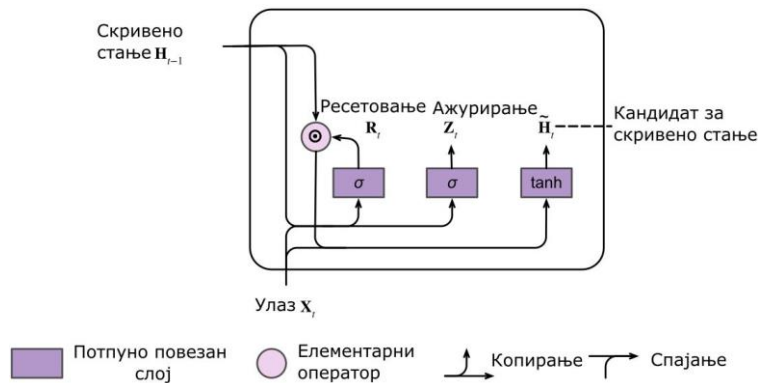
где су $\mathbf{W}_{xh} \in \mathbb{R}^{d \times h}$, $\mathbf{W}_{hh} \in \mathbb{R}^{h \times h}$ тежински коефицијенти, \mathbf{b}_h је слободни коефицијент, и \odot је Адамаров (франц. Hadamard) односно елементарни оператор производа. Овде се користи нелинеарност тангентне хиперболичне функције да би вредности кандидата за скривено стање остале у интервалу $(-1, 1)$. Резултат је само кандидат, јер тек треба интегрисати механизам за ажурирање. У односу на обичну рекурентну јединицу, код затворене рекурентне јединице, утицај претходних стања може бити редукован елементарним производом \mathbf{R}_t и \mathbf{H}_{t-1} . Када је вредност за ресетовање близу јединице, тада користимо стандардни приступ као код обичних рекурентних јединица. Са друге стране, када је вредност за ресетовање близу нуле, кандидат за скривено стање је резултат вишеслојног перцептрона са улазом \mathbf{X}_t .

5.10.1.3 Скривено стање

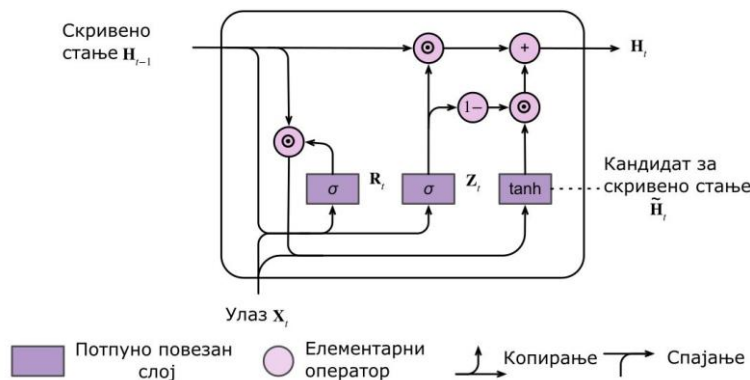
Да би ефекат ажурирања био инкорпориран у рекурентну јединицу, уводимо \mathbf{Z}_t . Овим се одлучује који удео скривеног стања \mathbf{H}_t је само копија старог стања \mathbf{H}_{t-1} и који удео новог кандидата за скривено стање $\tilde{\mathbf{H}}_t$ треба бити употребљен. Једначина за ажурирање стања у затвореној рекурентној јединици је:

$$\mathbf{H}_t = \mathbf{Z}_t \odot \mathbf{H}_{t-1} + (1 - \mathbf{Z}_t) \odot \tilde{\mathbf{H}}_t \quad (5.85)$$

Када механизам за ажурирање даје вредност близу 1, задржавамо старо стање. У овом случају, информација из тренутног корака је практично игнорисана. Са друге стране, ако је вредност близу нуле, ново латентно стање се приближава кандидату латентног стања. Овај дизајн помаже избегавању проблема нестајућих градијената код рекурентних мрежа.



Слика 5.23 Кандидат за скривено стање затворене рекурентне јединице [44]



Слика 5.24 Скривено стање затворене рекурентне јединице [44]

5.10.2 Јединица са дуготрајним-краткотрајним памћењем

Дизајн јединице са дуготрајним-краткотрајним памћењем је инспирисан логичким колима у рачунару. Јединица са дуготрајним-краткотрајним памћењем (енгл. LSTM) има слична својства као затворена јединица, али је комплекснија иако је настала две деценије пре ње [64, 65]. LSTM има меморијску јединицу, или ћелију, која је дизајнирана да чува додатне информације. За контролу меморијске јединице користи се неколицина механизма односно кола. Једно коло, које називамо излазним, је потребно да читава вредност из меморијске ћелије. Друго коло је потребно да уписује податке у меморијску ћелију. Ово коло ћемо звати улазним. Треће коло, засновано на механизму заборављања, је потребно да омогући ресетовање садржаја меморијске ћелије. Мотивација, која стоји иза дизајна LSTM, је слична као мотивација за дизајнирање затворене рекурентне јединице – неуронска мрежа треба да одлучи шта да запамти и које улазе да игнорише у скривеном стању.

5.10.2.1 Механизми за улаз, заборављање и излаз

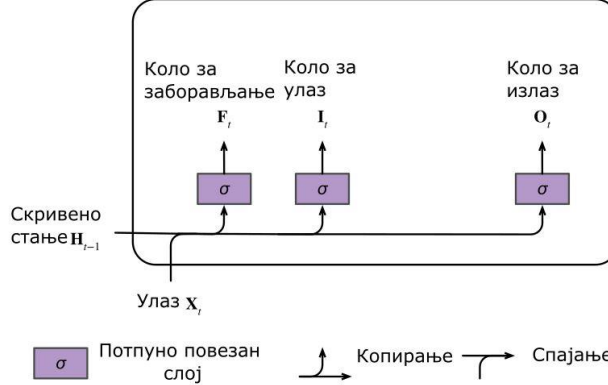
Улаз за тренутни временски корак и скривено стање из претходног временског корака пролазе кроз три потпуно повезана слоја са сигмоидалном активационом функцијом и тако процесирани се користе за улазно, излазно и коло за заборављање. Излази ових кола су вредности у опсегу (0,1). Нека имамо h скривених јединица, величину улаза n , број улазних атрибута d , улаз $\mathbf{X}_t \in \mathbb{R}^{n \times d}$ и скривено стање из претходног корака $\mathbf{H}_{t-1} \in \mathbb{R}^{n \times h}$, тада се улазни механизам $\mathbf{I}_t \in \mathbb{R}^{n \times h}$, механизам за заборављање $\mathbf{F}_t \in \mathbb{R}^{n \times h}$ и излазни механизам $\mathbf{O}_t \in \mathbb{R}^{n \times h}$ срачунавају формулама:

$$\mathbf{I}_t = \sigma(\mathbf{X}_t \mathbf{W}_{xi} + \mathbf{H}_{t-1} \mathbf{W}_{hi} + \mathbf{b}_i) \quad (5.86)$$

$$\mathbf{F}_t = \sigma(\mathbf{X}_t \mathbf{W}_{xf} + \mathbf{H}_{t-1} \mathbf{W}_{hf} + \mathbf{b}_f) \quad (5.87)$$

$$\mathbf{O}_t = \sigma(\mathbf{X}_t \mathbf{W}_{xo} + \mathbf{H}_{t-1} \mathbf{W}_{ho} + \mathbf{b}_o) \quad (5.88)$$

где су $\mathbf{W}_{xi}, \mathbf{W}_{xf}, \mathbf{W}_{xo} \in \mathbb{R}^{d \times h}$ и $\mathbf{W}_{hi}, \mathbf{W}_{hf}, \mathbf{W}_{ho} \in \mathbb{R}^{h \times h}$ тежински коефицијенти, а $\mathbf{b}_i, \mathbf{b}_f, \mathbf{b}_o \in \mathbb{R}^{1 \times h}$ су слободни коефицијенти. На слици 5.25 је дат шематски приказ кола за улаз, заборављање и излаз.



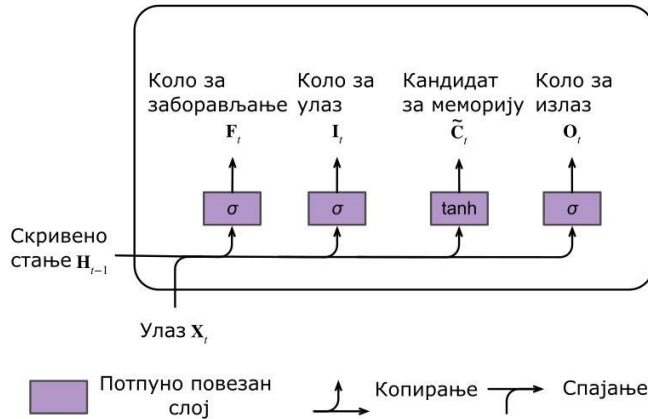
Слика 5.25 Кола за улаз, заборављање и излаз LSTM јединице [44]

5.10.2.2 Кандидат за меморијску јединицу

Кандидат за меморијску јединицу $\tilde{\mathbf{C}}_t$ срачунава се коришћењем хиперболичне тангентне функције:

$$\tilde{\mathbf{C}}_t = \tanh(\mathbf{X}_t \mathbf{W}_{xc} + \mathbf{H}_{t-1} \mathbf{W}_{hc} + \mathbf{b}_c) \quad (5.89)$$

где су $\mathbf{W}_{xc} \in \mathbb{R}^{d \times h}$ и $\mathbf{W}_{hc} \in \mathbb{R}^{h \times h}$ тежински коефицијенти и $\mathbf{b}_c \in \mathbb{R}^{1 \times h}$ су слободни коефицијенти. Илустрација кандидата за меморијску јединицу је дата на слици 5.26.



Слика 5.26 Кандидат за меморијску јединицу LSTM јединице [44]

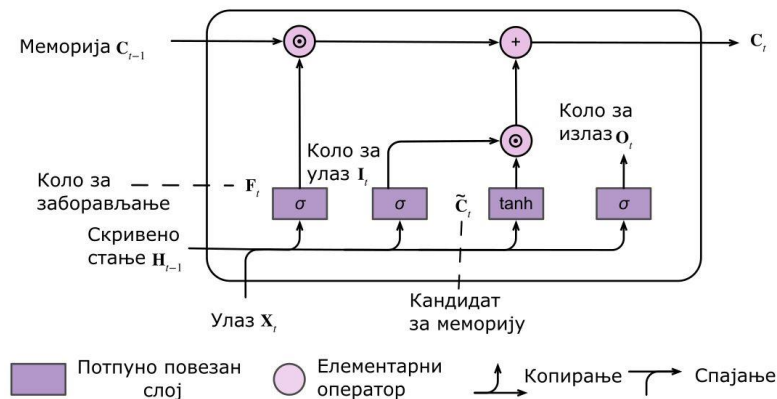
5.10.2.3 Меморијска јединица

Коло за улаз \mathbf{I}_t контролише коју удео улаза се узима у обзир преко кандидата за меморијску јединицу $\tilde{\mathbf{C}}_t$, а коло за заборављање \mathbf{F}_t контролише који удео старог садржаја меморијске јединице \mathbf{C}_{t-1} се узима у обзир. Користећи елементарни производ добија се једначина:

$$\mathbf{C}_t = \mathbf{F}_t \odot \mathbf{C}_{t-1} + \mathbf{I}_t \odot \tilde{\mathbf{C}}_t \quad (5.90)$$

Ако је вредност кола за заборављање приближно један и ако је вредност улазног кола приближно нула, стари садржај меморијске ћелије \mathbf{C}_{t-1} ће бити сачуван и коришћен за

тренутни временски корак. Шематски приказ срачунавања меморијске јединице дат је на слици 5.27.



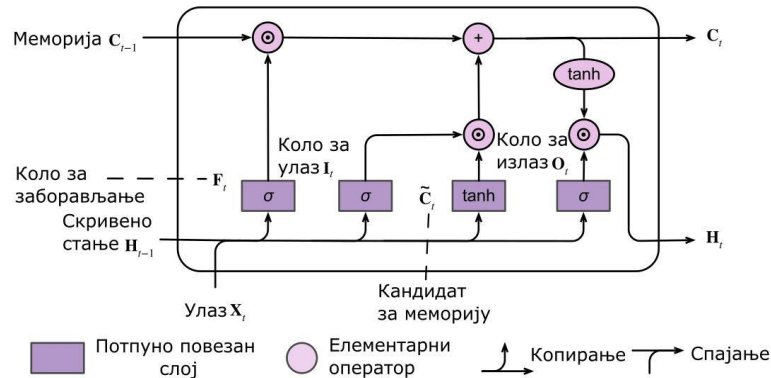
Слика 5.27 Меморија LSTM јединице [44]

5.10.2.4 Скривено стање

За рачунање скривеног стања користимо излаз излазног кола:

$$H_t = O_t \odot \tanh(C_t) \tag{5.91}$$

Када је вредност излазног кола приближно 1, преносимо информацију из меморије даље, а ако је вредност излазног кола приближно нула задржавамо информацију унутар меморијске јединице и не преносимо је даље. Хиперболична тангентна функција служи да се опсег вредности скривеног стања ограничи на опсег (-1,1). Срачунавање скривеног стања је приказано на слици 5.28.



Слика 5.28 Скривено стање LSTM јединице [44]

5.10.3 Угњеждена јединица са дуготрајном-краткотрајном меморијом

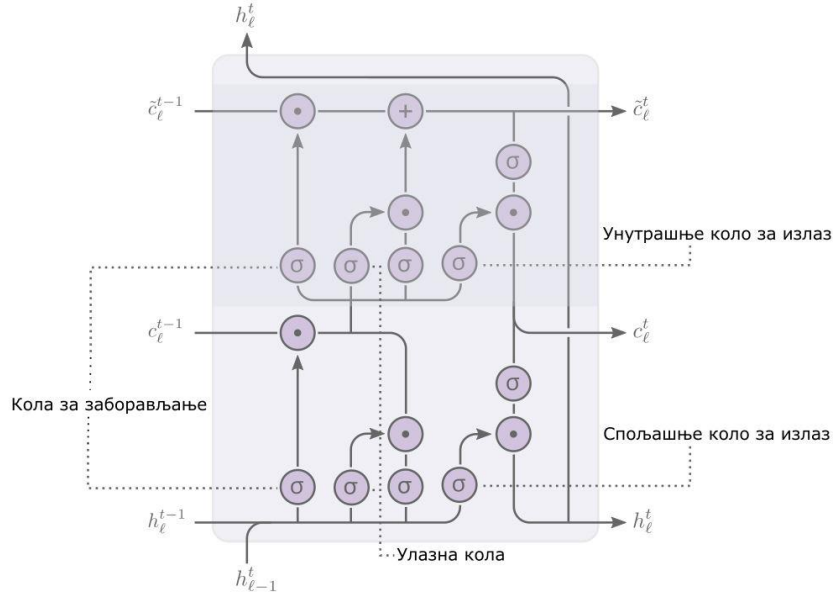
Док стандардне јединице са дуготрајном-краткотрајном меморијом (енгл. *LSTM*) решавају проблем додавањем интерне меморије [66], а затворене рекурентне јединице (енгл. *GRU*) представљају брже решење од јединица са дуготрајном-краткотрајном меморијом јер немају интерну меморију [66], угњеждене јединице са дуготрајном-краткотрајном меморијом (енгл. *nested LSTM*) представљају предлог решења нестајућих градијената, које иде у супротном правцу од затворене рекурентне јединице, јер се додаје више меморије у јединицу. Главна идеја је да додатна меморије у јединици омогућава бољу дугорочну меморију. Код угњеждених јединица са дуготрајном-краткотрајном меморијом операција додавања, која се користи за срачунавање c_t , је

замењена функцијом $c_t = m_t(f_t \odot c_{t-1}, i_t \odot g_t)$. Стање меморијске функције m у тренутку t је унутрашња меморија, која је моделирана јединицом са дуготрајном-краткотрајном меморијом, чиме се добија угњеждена јединица. Ова функција може бити и друга угњеждена јединица, чиме је омогућено дубоко угњеждавање. Улаз и скривена стања у меморијској функцији угњеждене јединице постају:

$$\tilde{h}_{t-1} = f_t \odot c_{t-1} \quad (5.92)$$

$$\tilde{x}_t = i_t \odot \sigma_c(x_t W_{xc} + h_{t-1} W_{hc} + b_c) \quad (5.93)$$

На слици 5.29 је дат шематски приказ угњеждене јединице са дуготрајном-краткотрајном меморијом. Словом l је обележен индекс скривеног слоја у ком се јединица налази.



Слика 5.29 Угњеждена јединица са дуготрајном-краткотрајном меморијом

5.10.4 Дубоке рекурентне неуронске мреже

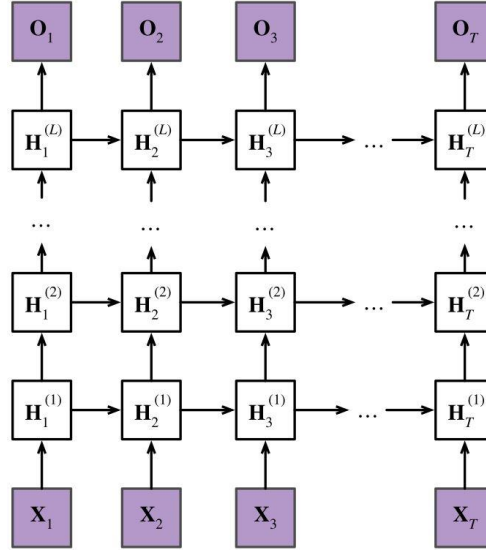
До сада смо разматрали рекурентне мреже са једним скривеним слојем. Као и код вишеслојног перцептрона, можемо паковати више рекурентних слојева један након другог. На слици 5.30 је шематски приказ дубоке рекурентне мреже са L слојева. Свако скривено стање се континуално преноси на наредни временски корак тренутног слоја и на исти временски корак наредног слоја. Нека имамо део података $\mathbf{X}_t \in \mathbb{R}^{n \times d}$ (n је број узорака, а d број улазних атрибута), нека је у кораку t скривено стање l -тог скривеног слоја $\mathbf{H}_t^{(l)} \in \mathbb{R}^{n \times h}$ (h је број неурона у скривеном слоју) и нека је $\mathbf{O}_t \in \mathbb{R}^{n \times q}$ излазни слој (q је број излаза из мреже). Постављањем $\mathbf{H}_t^{(0)} = \mathbf{X}_t$ скривено стање l -тог скривеног слоја који користи активациону функцију ϕ_l се формулише:

$$\mathbf{H}_t^{(l)} = \phi_l(\mathbf{H}_t^{(l-1)} \mathbf{W}_{xh}^{(l)} + \mathbf{H}_{t-1}^{(l)} \mathbf{W}_{hh}^{(l)} + \mathbf{b}_h^{(l)}) \quad (5.94)$$

за $l = 1, \dots, L$, где су $\mathbf{W}_{xh}^{(l)} \in \mathbb{R}^{h \times d}$ и $\mathbf{W}_{hh}^{(l)} \in \mathbb{R}^{h \times h}$ тежински коефицијенти, који заједно са слободним коефицијентима $\mathbf{b}_h^{(l)} \in \mathbb{R}^{1 \times h}$ представљају параметре l -тог скривеног слоја. Излазни слој зависи само од скривеног стања последњег скривеног слоја:

$$\mathbf{O}_t = \mathbf{H}_t^{(L)} \mathbf{W}_{hq} + \mathbf{b}_q \quad (5.95)$$

где су $\mathbf{W}_{hq} \in \mathbb{R}^{h \times q}$ тежински коефицијенти, који заједно са слободним коефицијентима $\mathbf{b}_q \in \mathbb{R}^{1 \times q}$ чине параметре излазног слоја. Као и код вишеслојног перцептрона, број скривених слојева и број јединица тј. неурона по слоју представљају хиперпараметре. Поред обичних рекурентних јединица, можемо користити и друге софистицираније рекурентне јединице, одговарајућом заменом формулације скривеног стања.



Слика 5.30 Шематски приказ дубоке рекурентне неуронске мреже

5.10.5 Бидирекционе рекурентне мреже

Некад имамо такав тип проблема, где је потребно гледати и информације које следе након тренутног временског корака. Уместо да користимо рекурентне мреже за пролаз кроз временске кораке од претходним ка наредним, можемо увести и још један правац, којим бисмо пролазили од последњег временског корака ка почетном. Бидирекционе рекурентне мреже уводе скривени слој који прослеђује информације уназад, зарад флексибилнијег процесирања информација. На слици 5.31 је приказана бидирекциона рекурентна мрежа са једним скривеним слојем. За дати корак t са делом улазних података $\mathbf{X}_t \in \mathbb{R}^{n \times d}$ (n је број узорака, а d број улазних атрибута) и скривеним слојем са активационом функцијом ϕ , скривено стање за процесирање унапред $\vec{\mathbf{H}}_t \in \mathbb{R}^{n \times h}$ и скривено стање за процесирање уназад $\overleftarrow{\mathbf{H}}_t \in \mathbb{R}^{n \times h}$ се дефинишу као:

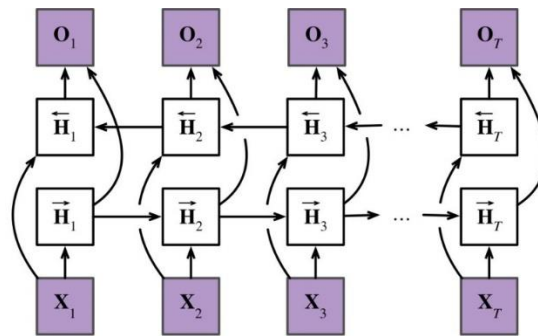
$$\vec{\mathbf{H}}_t = \phi(\mathbf{X}_t \mathbf{W}_{xh}^{(f)} + \vec{\mathbf{H}}_{t-1} \mathbf{W}_{hh}^{(f)} + \mathbf{b}_h^{(f)}) \quad (5.96)$$

$$\overleftarrow{\mathbf{H}}_t = \phi(\mathbf{X}_t \mathbf{W}_{xh}^{(b)} + \overleftarrow{\mathbf{H}}_{t+1} \mathbf{W}_{hh}^{(b)} + \mathbf{b}_h^{(b)}) \quad (5.97)$$

где је h број неурона, $\mathbf{W}_{xh}^{(f)} \in \mathbb{R}^{d \times h}$, $\mathbf{W}_{hh}^{(f)} \in \mathbb{R}^{h \times h}$, $\mathbf{W}_{xh}^{(b)} \in \mathbb{R}^{d \times h}$, $\mathbf{W}_{hh}^{(b)} \in \mathbb{R}^{h \times h}$ су тежински коефицијенти и $\mathbf{b}_h^{(f)}$, $\mathbf{b}_h^{(b)} \in \mathbb{R}^{1 \times h}$ су слободни коефицијенти. Спајањем скривених стања $\vec{\mathbf{H}}_t$ и $\overleftarrow{\mathbf{H}}_t$ добијамо скривено стање $\mathbf{H}_t \in \mathbb{R}^{n \times 2h}$ које треба проследити наредном скривеном слоју или излазном слоју. Излаз из излазног слоја $\mathbf{O}_t \in \mathbb{R}^{n \times q}$ се рачуна по формули:

$$\mathbf{O}_t = \mathbf{H}_t \mathbf{W}_{hq} + \mathbf{b}_q \quad (5.98)$$

где су $\mathbf{W}_{hq} \in \mathbb{R}^{2h \times q}$ тежински коефицијенти, $\mathbf{b}_q \in \mathbb{R}^{1 \times q}$ слободни коефицијент и q број излаза мреже. Два правца могу имати и различит број скривених неурона.



Слика 5.31 Шематски приказ бидирекционе рекурентне неуронске мреже

Једна од кључних карактеристика бидирекционе рекурентне мреже је да се користе информације са оба краја секвенце за предикцију излаза. Другим речима користе се информације и из будућности и из прошлости да би се предвидело тренутно стање. У току обуке се могу имати и подаци пре и након посматраног тренутка, док у току тестирања можемо имати и само претходне временске кораке, што некад доводи до лоших перформанси. Бидирекционе рекурентне мреже су спорије него обичне рекурентне мреже. Главни разлог за ово је што пропација унапред захтева рекурентне операције за оба смера у бидирекционим слојевима, а пропација уназад је зависна од резултата пропације унапред, па ће градијенти имати веома дугачке ланце зависности. У пракси се бидирекциони слојеви ретко користе. Уобичајан пример коришћења би било допуњавање речи које недостају у реченици.

5.11 Конволуционе неуронске мреже

Неуронске мреже су инваријантне у односу на редослед улазних атрибута тако да можемо добити сличне резултате без обзира на њихов редослед. Када имамо неку просторну структуру као што су слике, било би добро да можемо да искористимо информацију да су суседни пиксели у неком међусобном односу. У овој секцији уводимо конволуционе неуронске мреже, које су настале баш зарад процесирања информација код којих постоји просторна зависност. Овај тип неуронских мрежа се најчешће користи у проблемима рачунарског вида. Конволуционе мреже су рачунски ефикасне, захтевају мање параметара него потпуно повезани слојеви код вишеслојног перцептрона, а процес конволуције је врло погодан за паралелизацију на графичким процесорским јединицама.

5.11.1 Инваријантност код конволуционих мрежа

Замислимо да је потребно препознати објекат на слици. У том случају није неопходно да знамо тачан положај објеката, јер изглед објекта нема везе са тим где се он налази у простору. Конволуционе неуронске мреже систематизују идеју просторне инваријантности и користе је у току учења репрезентације објекта, користећи мањи број параметара него стандардни вишеслојни перцептрон. Можемо посматрати вишеслојни перцептрон са дводимензионалним сликама као улазом \mathbf{X} и скривеним међуслојем \mathbf{H} који такође представљамо као дводимензионалну матрицу. Нека скривени слој \mathbf{H} има исте димензије као улаз, или другим речима нека скривени слој представља скривену репрезентацију улазне слике. Улаз у мрежу, као и скривене репрезентације улаза унутар мреже, имају просторну структуру. Нека су $[\mathbf{X}]_{i,j}$ и

$[H]_{i,j}$ пиксели на локацији (i, j) у улазној слици и у њеној скривеној репрезентацији. Потпуно повезан слој H можемо тада изразити као:

$$[H]_{i,j} = [U]_{i,j} + \sum_k \sum_l [W]_{i,j,k,l} [X]_{k,l} = [U]_{i,j} + \sum_a \sum_b [V]_{i,j,a,b} [X]_{i+a,j+b} \quad (5.99)$$

при чему су W тежински коефицијенти, а U садржи слободне коефицијенте. Прелазак са нотације W на U је чисто козметичке природе јер постоји један на један кореспонденција између коефицијената у оба тензора. Индекси (k, l) су такви да је $k = i + a$ и $l = j + b$, тј важи да је $[V]_{i,j,a,b} = [W]_{i,j,i+a,j+b}$. За било коју дату локацију (i, j) у скривеној репрезентацији, рачунамо вредност $[H]_{i,j}$ сумирањем пиксела по x , са центром (i, j) и примењујемо тежинске коефицијенте $[V]_{i,j,a,b}$. Ако имамо 1000×1000 слику, имаћемо и њену скривену репрезентацију исте величине, што значи да имамо 10^{12} параметара, што је веома велики број. Да бисмо смањили број параметара, применићемо транслациону инваријантност. Померај у улазу треба да доведе до помераја у скривеној репрезентацији. Ово је могуће само у случају да V и U не зависе од (i, j) , што значи да имамо $[V]_{i,j,a,b} = [V]_{a,b}$ а U је само константа коју ћемо обележити са u . Имајући претходно наведено у виду, можемо упростити дефиницију H :

$$[H]_{i,j} = u + \sum_a \sum_b [V]_{a,b} [X]_{i+a,j+b} \quad (5.100)$$

Претходна формула представља конволуцију која ће бити детаљније појашњена као процес у наредној подсекцији. Треба приметити да је број коефицијената код $[V]_{a,b}$ мањи него код $[V]_{i,j,a,b}$, јер више није потребна локација слике. Последнично број параметара више није 10^{12} , него $4 * 10^6$.

5.11.2 Локалност код конволуционих мрежа

Не треба ићи предалеко од локације (i, j) да бисмо обухватили релевантне информације за $[H]_{i,j}$. То значи да изван неког опсега $|a| > \Delta$ или $|b| > \Delta$ треба да буде $[V]_{a,b} = 0$. Можемо преформулисати H као:

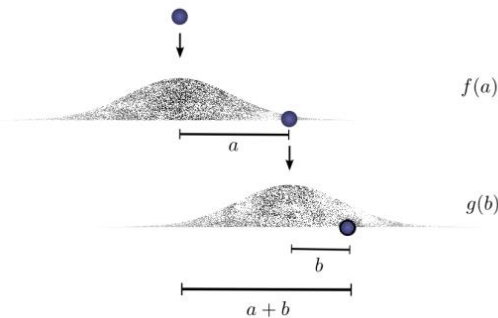
$$[H]_{i,j} = u + \sum_{a=-\Delta}^{\Delta} \sum_{b=-\Delta}^{\Delta} [V]_{a,b} [X]_{i+a,j+b} \quad (5.101)$$

Ово редукује број параметара са $4 * 10^6$ на $4 * \Delta^2$, где је Δ типично мање од 10. Цена коју смо платили за овако значајно смањење параметара је да једним скривеним слојем обухватамо само локалне информације и да је потребно да улазни атрибути буду транслаторно инваријантни. Како идемо кроз конволуциону мрежу, дубљи слојеви треба да представљају комплексније и веће аспекте слике. Ово се постиже наизменичним преплитањем нелинеарности и додатних конволуционих слојева.

5.11.3 Операција конволуције

Да бисмо разумели конволуционе неуронске мреже, треба објаснити конволуцију. Посматрајмо испуштање лоптице са неке висине и њену позицију након испуштања. Која је вероватноћа да ће лоптица бити на растојању c у односу на почетни положај, ако је испустимо са неке висине, а затим поново подигнемо и испустимо? После првог испуштања, лоптица ће слетети a метара од почетне позиције са вероватноћом $f(a)$. Подижемо лоптицу са позиције a и испуштамо је са друге висине. Вероватноћа да је лоптица на растојању b од позиције a је $g(b)$, где g може бити другачија дистрибуција

од f , с обзиром да лоптица може бити испуштена са друге висине. Процес је приказан на слици 5.32.

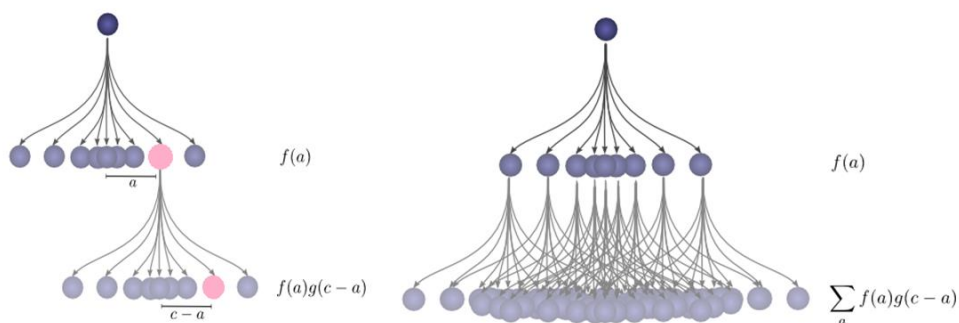


Слика 5.32 Испуштање лоптице [67]

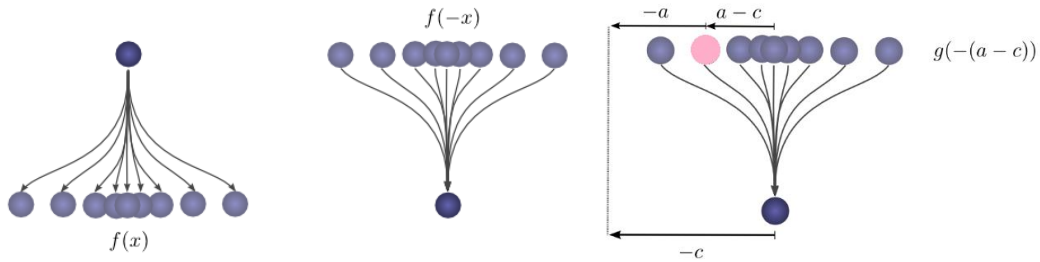
Ако узмемо $c = a + b$, имамо да је вероватноћа да се лоптица нађе на растојању c у односу на почетни положај, после другог испуштања, једнака $f(a)g(b)$. Ако је, на пример $c = 3$, тада можемо да узмемо $a = 2, b = 1$ или $a = 1, b = 2$ или $a = 0, b = 3$ или $a = 3, b = 0$. Да бисмо нашли укупну вероватноћу да је лоптица на растојању c у односу на почетни положај, после другог бацања, морамо да узмемо у обзир све начине на које се стиже до растојања c : $\sum_{a+b=c} f(a)g(b)$. Ово је конволуција. Формално конволуција функција f, g у тачки c се дефинише као:

$$(f * g)(c) = \sum_{a+b=c} f(a)g(b) = \sum_a f(a)g(c - a) \tag{5.102}$$

Шематски приказ конволуције на примеру испуштања лоптице је дат на слици 5.33. Ако је $f(x)$ вероватноћа да је лоптица на позицији x после испуштања, тада је $f(-x)$ вероватноћа да је лоптица била на позицији x пре испуштања (наредно бацање). Ако је познато да је лоптица на позицији c после другог испуштања, која је вероватноћа да је преходно била на позицији a ? На основу слике 5.34 види се да је та вероватноћа $g(-(a - c)) = g(c - a)$.



Слика 5.33 Визуализација конволуције на примеру испуштања лоптице [67]



Слика 5.34 Шематски приказ одређивања вероватноће на примеру испуштања лоптице [67]

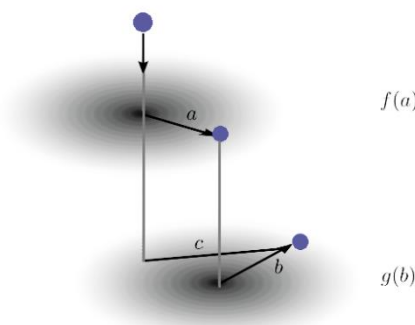
Конволуција је општа идеја, и може да се користи и у проблемима са више димензија. Може се посматрати пример лоптице, где се при паду посматрају x и y координате лоптице (Слика 5.35). Конволуција је иста као пре, само су сада \mathbf{a} , \mathbf{b} и \mathbf{c} вектори:

$$(f * g)(\mathbf{c}) = (f * g)(c_1, c_2) = \sum_{\mathbf{a}_1, \mathbf{a}_2} f(\mathbf{a}_1, \mathbf{a}_2)g(\mathbf{c}_1 - \mathbf{a}_1, \mathbf{c}_2 - \mathbf{a}_2) = \sum_{\mathbf{a}} f(\mathbf{a})g(\mathbf{c} - \mathbf{a}) \quad (5.103)$$

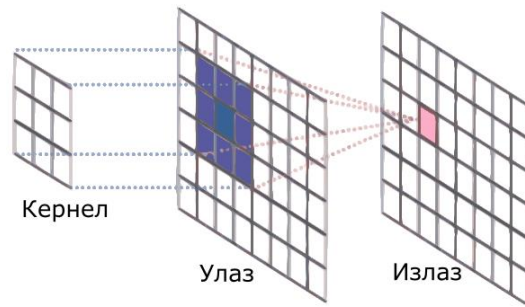
Као и са једном димензијом, вишедимензионалну конволуцију можемо да посматрамо као “клизање” једне функције преко друге, где их множимо и сабирамо резултате. Једна од типичних примена конволуције је процесирање слика. Сliku можемо посматрати као дводимензионалну функцију. Многе важне трансформације се одвијају тако што се на слику примењује мала локална функција, коју често називамо кернелом. Кернел клизи преко слике и рачуна вредност новог пиксела. Ова вредност се рачуна као тежинска сума пиксела преко којих је кернел прешао у том кораку. Шематски приказ је дат на слици 5.36. Кернели могу имати различите величине. Поставља се питање шта се дешава када кернел излази ван делова слика. Некад се ти делови једноставно игноришу, или се матрицама додају нуле на крајевима врста и колона. Број додатих колона односно врста се у терминологији често назива *padding*. Број пиксела за који се кернел помера у току конволуције се назива *stride*. Уз помоћ задатих вредности за *padding* (P) и *stride* (S) може се прорачунати величина излаза:

$$O = \frac{W-K+2P}{S} + 1 \quad (5.104)$$

где је O излазна димензија (ширина, висина), W улазна димензија, а K величина кернела.



Слика 5.35 Испуштање лоптице у две посматране димензије [67]



Слика 5.36 Примена конволуције на слику

5.11.4 Конволуциони слојеви

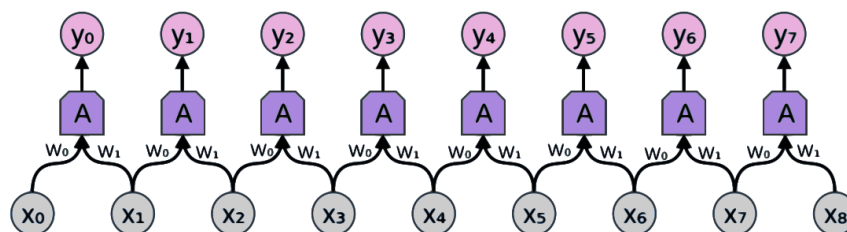
О конволуционој неуронској мрежи можемо да размишљамо као о мрежи која користи многе идентичне копије истог неурона. Ово омогућава мрежи да има много неурона, а да број параметара, вредности које описују како се неурони понашају, буде мали. Нека имамо 1-Д конволуциони слој са улазима $\{x_n\}$ и излазима $\{y_n\}$. Шематски приказ овог слоја дат је на слици 5.37. Изразе изражавамо у зависности од улаза: $y_n = A(x_n, x_{n+1}, \dots)$. Генерално A може бити више неурона, али ћемо тренутно посматрати као да је један. Понашање типичног неурона у мрежи је описано једначином:

$$\sigma(\omega_0 x_0 + \omega_1 x_1 + \dots + b) \tag{5.105}$$

где су x_i улази, ω_i тежински коефицијенти. Два неурона су једнака ако су им тежински коефицијенти једнаки. Конволуција ће одредити тежинске коефицијенте неурона и одредиће који су неурони једнаки. Типично се понашање једног слоја мреже описује једначином:

$$y = \sigma(Wz + b) \tag{5.106}$$

где је W матрица тежинских коефицијената. Пошто постоји више копија истог неурона, многи тежински коефицијенти ће се појављивати више пута у матрици. Како неурони нису повезани са свим улазима, тежинска матрица за конволуциони слој ће имати пуно нула.



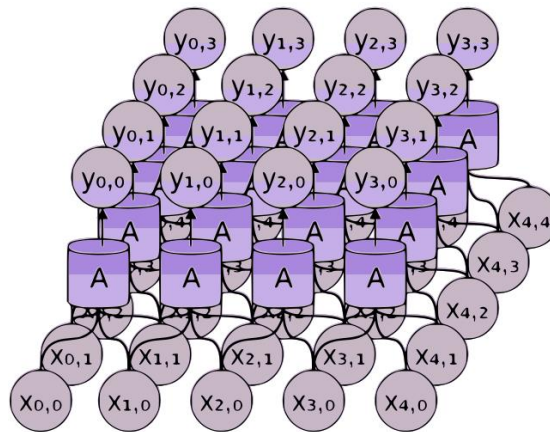
Слика 5.37 Приказ једнодимензионалног конволуционог слоја

Шематски приказ неуронске мреже са 2-Д конволуционим слојем дат је на слици 5.38. До сада смо игнорисали чињеницу да слике имају три канала: црвени, зелени и плави. Сlike нису заиста дводимензионални објекти већ тензори трећег реда, са димензијама које означавају висину, ширину и канале (нпр. $1024 \times 1024 \times 3$). Дакле, улаз X можемо да обележимо као $[X]_{i,j,k}$, а конволуциони филтер треба прилагодити аналогно - уместо $[V]_{a,b}$ имамо $[V]_{a,b,c}$. Пошто је улаз тензор трећег реда, треба и скривене слојеве

формулисати као тензоре трећег реда. Уместо да имамо једну скривену репрезентацију која одговара свим просторним локацијама, имаћемо вектор скривених репрезентација за сваку просторну локацију. Некада се ове скривене репрезентације називају мапама атрибута. Да бисмо имали могућност да радимо са више канала у улазу (\mathbf{X}) и више канала у скривеним репрезентацијама (\mathbf{H}) додајемо четврту координату у \mathbf{V} : $[\mathbf{V}]_{a,b,c,d}$. Дакле, имамо:

$$[\mathbf{H}]_{i,j,d} = \sum_{a=-\Delta}^{\Delta} \sum_{b=-\Delta}^{\Delta} \sum_c [\mathbf{V}]_{a,b,c,d} [\mathbf{X}]_{i+a,j+b,c} \quad (5.107)$$

где су d индекси излазних канала у скривеној репрезентацији \mathbf{H} . Наредни конволуциони слој ће преузети излазни тензор из \mathbf{H} као улаз.



Слика 5.38 Дводимензионални конволуциони слој

Као и са вишеслојним перцептроном, можемо имати више конволуционих слојева. Са више слојева можемо детектовати апстрактније карактеристике. Често се уз конволуционе слојеве додају и агрегациони (енгл. *pooling*) слојеви, од којих посебно можемо издвојити слој за налажење максимума (енгл. *max-pooling*). Може се рећи да је улога ових слојева зумирање. Ови слојеви замењују делове излаза претходног слоја једном вредношћу. У случају “*max-pooling*” слоја, та вредност је максимум. Ово може бити веома корисно за уочавање неког шаблона на вишем нивоу. Нека је, на пример, улаз у неуронску мрежу звучни сигнал, који желимо да класификујемо на неки начин. Може бити од интереса да ли се јављају нагли прелазни у фреквенцији. Коришћењем слоја за агрегацију ово може бити раније уочено, јер ће он смањити величину улаза и истаћи бројеве. Овим би се изгубила тачна локација наглог прелазног фреквенције, али нам за класификацију највероватније и неће бити важно кад се тачно јавио прелаз, него да ли прелаз има.

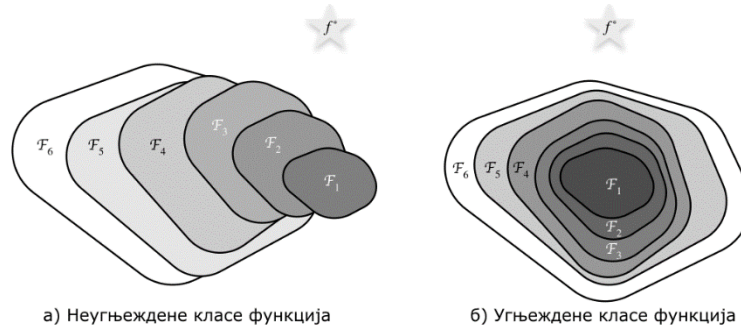
Конволуционе мреже имају примену у различитим областима. Мреже са једнодимензионалним конволуционим слојевима се најчешће користе за обраду звучног сигнала, на пример у препознавању говора. Мреже са дводимензионалним слојевима се најчешће користе за обраду слика, на пример у препознавању облика на сликама, а мреже са тродимензионалним конволуционим слојевима се најчешће користе за видео записе. Многи значајни резултати су постигнути коришћењем ових мрежа и оне су постале неопходан алат у препознавању шаблона. Треба напоменути да се при тренирању, делови мреже често специјализују. На пример, приликом обучавања

мреже за препознавање облика на сликама, један део мреже ће се фокусирати на детектовање ивица, други део на детектовање боје и текстуре, а трећи на облике.

5.11.5 Резидуалне мреже

Креирањем дубљих неуронских мрежа повећава се комплексност и експресивност мреже. Нека је F класа функција коју специфична архитектура мреже, заједно са другим хиперпараметрима, може да апроксимира. За сваку функцију $f \in F$ постоји неки сет параметара тј. тежинских и слободних коефицијената који се могу пронаћи алгоритмом обучавања и погодним скупом података за обуку. Нека је f^* права функција, коју треба наћи. Тражимо неку f_F^* из класе функција F , која је најприближнија правој функцији f^* . Нека је X улазни скуп података, и нека су y ознаке, тада можемо тражити функцију f_F^* решавањем следећег оптимизационог проблема:

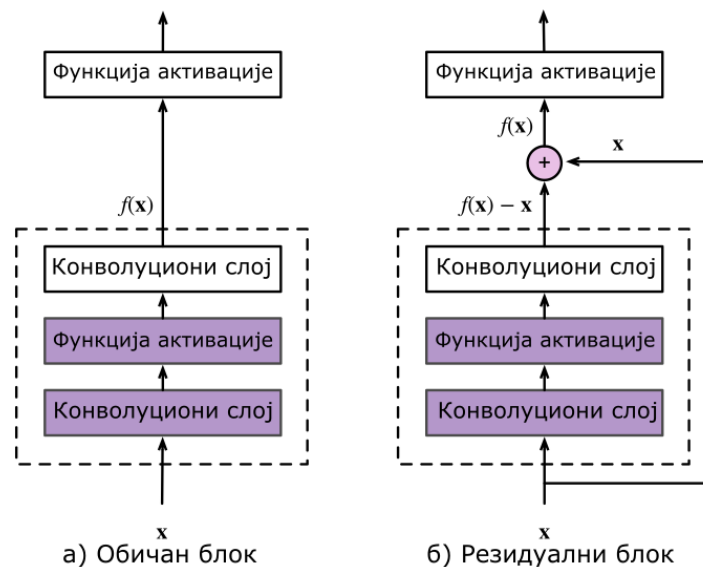
$$f_F^* \stackrel{\text{def}}{=} \underset{f}{\operatorname{argmin}} L(X, y, f), \text{ за } f \in F \quad (5.108)$$



Слика 5.39 Неугњежене и угњежене класе функција

Различите методе регуларизације могу да контролишу комплексност F , а већи скуп података за обучавање генерално води до боље f_F^* . Може се претпоставити да ако можемо да дизајнирамо другачију, моћнију архитектуру која може да апроксимира функције из F' , тада можемо и да пронађемо бољу апроксимацију $f_{F'}^*$ праве функције f^* . Другим речима очекујемо да је $f_{F'}^*$ боља апроксимација од f_F^* . Међутим, ако важи $F \not\subseteq F'$, нема гаранције да ће $f_{F'}^*$ бити боља апроксимација од f_F^* . За неугњежене класе функције, већа класа не мора да буде ближа правој функцији. На пример на слици 5.39, се види да иако је F_3 ближа f^* него F_1 , F_6 се удаљава од праве функције и нема гаранције да додатно повећавање комплексности може да редукује удаљеност од f^* . Са угњеженим класама функција, где је $F_1 \subseteq \dots \subseteq F_6$, се може избећи овај проблем. Дакле, само ако већа функција класа садржи мање, можемо гарантовати да повећавањем класа повећавамо и експресивну моћ мреже. За дубоку неуронску мрежу, тада важи да ако можемо да обучимо додати слој као функцију идентитета $f(x) = x$, нова мрежа ће бити једнако добра као оригинала мрежа без додатног слоја. Како нова мрежа има више слојева, она може да достигне боље решење које боље фитује скуп података за обуку. Посматрајемо локални део неуронске мреже на слици 5.40. Нека је x улаз, и нека је жељена функција коју треба научити $f(x)$. Део уоквирен испрекиданом линијом са леве стране на слици 5.40, треба да научи функцију $f(x)$ директно. Са десне стране део уоквирен испрекиданом линијом треба да научи резидуално мапирање

$f(x) - x$. По овом резидуалном мапирању, резидуалне мреже су и добиле своје име. Ако је идентично мапирање жељено мапирање, тада је резидуално мапирање лакше научити, јер тада сви тежински и слободни коефицијенти треба да имају вредност нула. Десни део слике 5.40 илуструје резидуални блок, где се пуна линија, која преноси улаз x до оператора адиције, назива резидуалном везом.



Слика 5.40 Обичан и резидуални блок

5.12 Конволуционе неуронске мреже за процесирање секвенцијалних података

Иако се најчешће користе за класификацију слика, конволуционе неуронске мреже су се показале као добар алат за анализу временских серија, уз одређене модификације [68,69]. У овој секцији ћемо описати детаље временских конволуционих мрежа (енгл. *Temporal Convolutional Network* или скраћено TCN). Рекурентне мреже се најчешће користе за анализу секвенцијалних података, међутим конволуционе мреже су лакше за обуку, јер нису толико подложне проблемима нестајућих и експлодирајућих градијената. Поред тога, конволуционе неуронске мреже су рачунски ефикасније и погодније су за паралелизацију. Градивне јединице временских конволуционих мрежа су проширени једнодимензионални конволуциони слојеви са истом величином улаза и излаза. Конволуција у временској конволуционој мрежи је каузална конволуција (енгл. *causal convolution*), што значи да не постоји цурење информација из будућности у прошлост. TCN могу да процесирају секвенце било које дужине и мапирају је на секвенцу исте дужине, као што то чине и рекурентне неуронске мреже. Нека је дата улазна секвенца x_0, \dots, x_T и нека треба предвидети одговарајуће излазе y_0, \dots, y_T . Кључно ограничење је да се за предвиђање y_t у тренутку t , могу користити само улази који претходе t : x_0, \dots, x_t . Формално, неуронска мрежа која моделира неку секвенцу је функција $f: X^{T+1} \rightarrow Y^{T+1}$ која даје мапирање:

$$\hat{y}_0, \dots, \hat{y}_T = f(x_0, \dots, x_T) \quad (5.109)$$

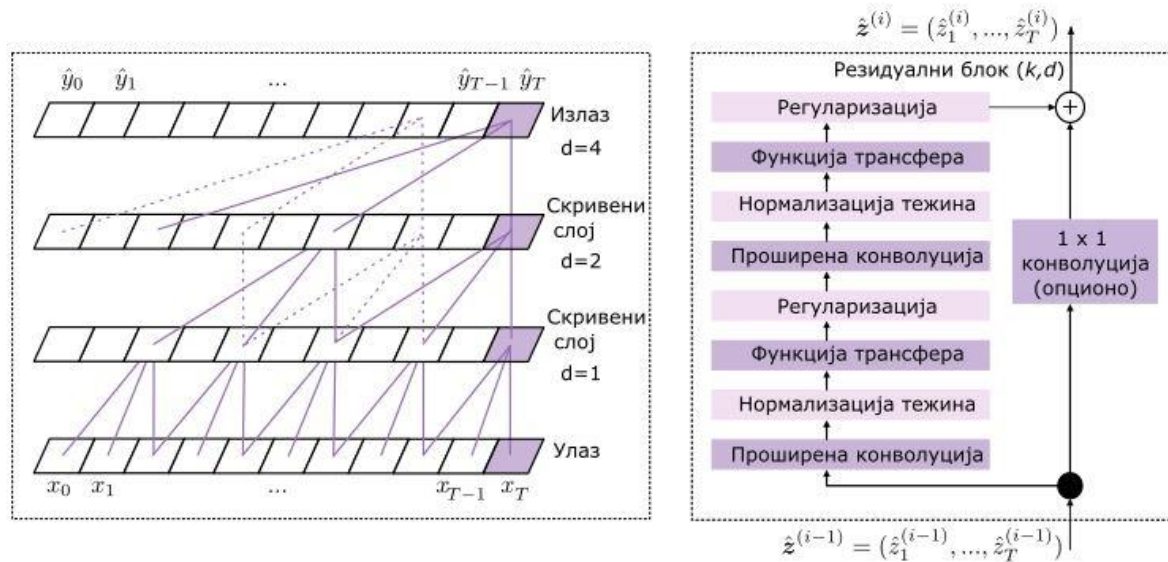
ако задовољава узрочно ограничење да y_t зависи само од x_0, \dots, x_t а не од неког од будућих улаза x_{t+1}, \dots, x_T . Циљ обуке је наћи функцију која минимизује функцију

губитка $L(y_0, \dots, y_T, f(x_0, \dots, x_T))$ која представља разлику стварних и предвиђених излаза. Овај формализам обухвата многе проблеме као што је на пример ауторегресивно предвиђање, где је потребно предвидети будуће понашање сигнала на основу његовог досадашњег понашања. У случају ауторегресивног предвиђања, у току обуке, циљани излаз је заправо улаз померен за један временски корак уназад. Ова дефиниција не обухвата машинско превођење, јер постоје случајеви где се цела улазна секвенца, укључујући и будућа стања и улазе, може користити за предикцију. Међутим, дефиниција се може проширити и на задатке тог типа.

Да би временска конволуциона мрежа давала излаз исте дужине као што је дужина улаза користе се једнодимензионални потпуно повезани конволуциони слојеви, где је сваки скривени слој исте величине као улазни слој, и проширен (енгл. *padding*) са (величина кернала $- 1$) нула да би сваки следећи слој задржао исту дужину. Једноставном каузалном конволуцијом можемо да посматрамо број корака који је пропорционалан дубини мреже. Број временских корака које можемо да обухватимо се назива рецептивним пољем временске конволуционе мреже. Да бисмо повећали рецептивно поље, и тиме олакшали употребу временских конволуционих мрежа у моделирању секвенци, уводимо проширену каузалну конволуцију. Формално, за једнодимензионални секвенцијални улаз $x \in \mathbb{R}^n$ и филтер $f: \{0, \dots, k-1\} \rightarrow \mathbb{R}$ проширени оператор конволуције F на елементу s дате секвенце се дефинише као:

$$F(s) = (x *_d f)(s) = \sum_{i=0}^{k-1} f(i) x_{s-di} \quad (5.110)$$

где је d фактор проширења (енгл. *dilation factor*), k је величина филтера, и $s - di$ се односи на претходне кораке који се посматрају. Повећавањем величине филтера и фактора проширења, повећава се величина рецептивног поља временске конволуционе мреже. Шематски приказ временске конволуционе мреже је дат на слици 5.41. Обично се фактор проширења повећава експоненцијално са дубином мреже нпр. $d = O(2^i)$ где i представља индекс слоја мреже. Често се у имплементацији временске конволуције користе и резидуални блокови зарад стабилизације када мрежа постане дубока. Уобучајена форма резидуалног блока за временску конволуциону мрежу је приказана на слици 5.41. Унутар резидуалног блока имамо два слоја проширене конволуције и уводимо нелинеарност функцијом активације односно функцијом трансфера. За функцију трансфера најчешће се користи ReLU, поред тога користи се нормализација тежина и додатна регуларизација, најчешће у виду просторног изостављања (енгл. *spatial dropout*). Док се код стандардних резидуалних мрежа, улаз додаје директно на излаз резидуалног блока, код временске конволуције излаз и улаз су различитих ширина па је потребно додати 1×1 конволуциони блок да би адициони оператор добио тензоре одговарајућих димензија.



Слика 5.41 Шематски приказ временске конволуционе неуронске мреже

Предности временских конволуционих мрежа су:

1. **Паралелизација.** Док се код рекурентних мрежа предикције морају вршити редом по временским корацима, код конволуционих мрежа ово може бити урађено паралелно јер сваки слој користи исти филтер. И у току обуке и у току евалуације, дуга улазна секвенца може бити процесирана у целости кроз конволуциону мрежу.
2. **Флексибилна величина рецептивног поља.** Величина рецептивног поља се може променити на више начина нпр. додавањем додатних слојева са већим фактором проширења, или без додавања слојева али коришћењем већих фактора проширења или повећавањем величине филтера. Временске конволуционе мреже тиме нуде већу контролу потрошње меморије и дају могућност адаптирања на различите домене.
3. **Стабилни градијенти.** За разлику од рекурентних мрежа, TCN користе пропацију уназад која нема везе са временским правцем секвенце, тиме избегавајући проблеме нестајућих и експлодирајућих градијената.
4. **Ниска потрошња меморије у току обуке.** За процесирање дугих секвенци, код GRU и LSTM мрежа је потребно чувати парцијалне резултате различитих кола. Пошто се код TCN мрежа користи исти филтер за цео слој, пропација уназад зависи само од дубине мреже.
5. **Променљива дужина улаза.** Слично као и код рекурентних мрежа и TCN може процесирати улазе различитих димензија.

Мане временских конволуционих мрежа су:

1. **Чување података у току евалуације.** У току евалуације, рекурентне мреже чувају само скривена стања и тренутни улаз зарад генерисања предикције. Другим речима, историја фиксне дужине је представљена скривеним стањем, и

може се изоставити из улазне секвенце. Насупрот томе TCN узима целу секвенцу.

- 2. Потенцијално мењање параметара приликом промене домена примене.** Другачији домени могу имати различите захтеве у погледу историје и редоследа предикције. Приликом трансфера модела из домена где је потребна мања количина меморије (мала величина филтера и мале вредности за факторе проширења), у домен где је потребно више меморије, перформансе TCN могу бити лоше због недовољно великог рецептивног поља.

6 Креирање сурогат модела

У овој секцији ће бити демонстриране процедуре за креирање сурогат модела. Можемо разликовати: (1) сурогат моделе вођене подацима (енгл. *data-driven*), засноване на неуронским мрежама за анализу временских серија и (2) сурогат моделе засноване на неуронским мрежама подржаним Хакслијевом једначином за мишићну контракцију.

За потребе креирања сурогат модела, прикупљени су подаци из нумеричких симулација, при чему се користе фиксни параметри за Хакслијев материјални модел мишића, који су дати у табели 6.1. Простор у ком се посматра миозинска глава је дефинисан параметрима $xstart$ (почетна координата), $xend$ (крајња координата) и $xdiv$ (параметар за дискретизацију односно поделу по “x”). У случају сурогат модела заснованог на неуронским мрежама за анализу временских серија коришћене су вредности $xstart = -23.4$, $xend = 327.6$, $xdiv = 22502$. За креирање сурогат модела заснованог на неуронским мрежама подржаним Хакслијевом једначином, коришћене су вредности $xstart = -2.08$, $xend = 20.28$, $xdiv = 45$. Разлог за коришћење мање поделе у случају неуронских мрежа подржаних Хакслијевом једначином је смањење величине улазног тензора. Овим се губи на прецизности, али се смањује потрошња меморије.

Табела 6.1 Параметри Хакслијевог модела мишића

<i>Параметар</i>	<i>Вредност</i>
Почетна дужина саркомере (L_0)	1100.0 [nm]
Површина попречног пресека саркомере (A)	130.0 [nm ²]
Крутоост попречних мостова (K)	0.58 [pN/nm]
Критична дужина попречног моста (h)	15.6 [nm]
Параметар стопе закачињања миозинске главе (f_1)	43.3 [s ⁻¹]
Параметар стопе откачињања миозинске главе (g_1)	10.0 [s ⁻¹]
Параметар стопе откачињања миозинске главе (g_2)	208.0 [s ⁻¹]

6.1 Креирање сурогат модела заснованог на неуронским мрежама за анализу временских серија

За креирање сурогат модела заснованог на неуронским мрежама за анализу временских серија се најпре користи генератор нумеричких експеримената за генерисање улазних фајлова у програм за анализу методом коначних елемената [20]. Генерисани модели садрже један 2Д коначни елемент, који је приказан поред генератора на слици 6.1, са граничним условима који варирају у зависности од типа експеримента. Разликују се 4 типа нумеричких експеримената: (1) изотонична контракција, (2) брзо ослобађање, (3) задате силе и (4) задата померања.

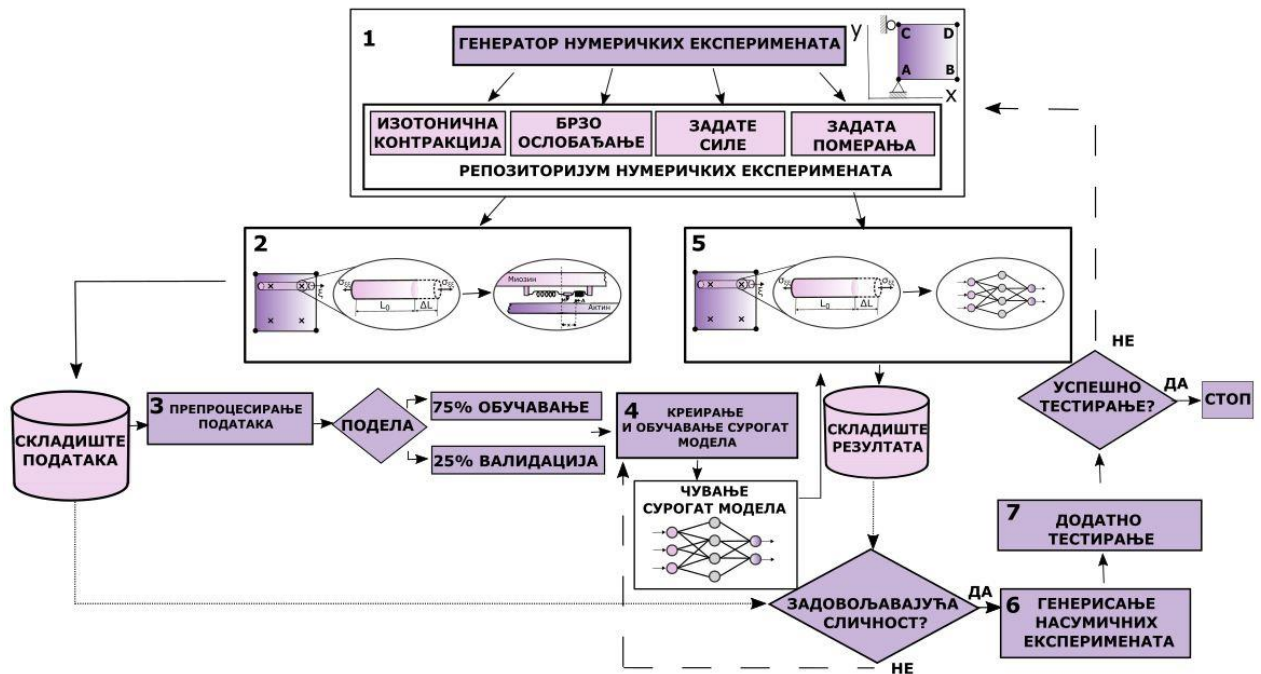
За експерименте са изотоничном контракцијом, translације су ограничене у чвору А у оба правца x и y , док су у тачки С translације ограничене у x -правцу. Генератор варира функцију активације односно концентрације калцијума за сваки генерисани

експеримент са изотоничном контракцијом. У току изотоничне контракције мишић се активира, директно функцијом активације или индиректно конвертовањем функције концентрације калцијума у активацију. Под утицајем активације мишић се контрахује, а затим се, када је деактивиран, полако враћа у првобитни положај. У току експеримената са брзим ослобађањем, мишић је потпуно активиран и транслације у свим правцима су ограничене у чворовима А, В, и D, све до специфицираног временског тренутка, када се транслациона ограничења уклањају у чворовима D и В, и када се задаје сила која је мања од 25% максималне генерисане силе. Генератор варира време у ком се мишић ослобађа као и вредност задате силе. За експерименте са задатим силама, силе се задају у чворовима D и В, док су транслациона ограничења иста као код изотоничних контракција. Генератор варира функцију задате силе. За експерименте са задатим померањима, померања се задају у чворовима D и В, док су транслациона ограничења иста као код изотоничних контракција. Генератор варира функцију задатих померања.

Након што су улази генерисани, покреће се *Mexie*, програм за анализу методом коначних елемената, ради сакупљања података. Чувају се мишићна активација, стречеви, напони и изводи напона, који су улази у неуронске мреже. Приликом сакупљања података на нивоу интеграционе тачке се користи оригинални Хакслијев модел за добијање напона и извода напона. Након прикупљања, следи препроцесирање података и конвертовање података у временске серије. Улазни тензор у неуронску мрежу за анализу временских серија је тродимензионалан, при чему су димензије редом једнаке (1) броју података у којима предвиђамо излазне вредности, (2) броју улазних атрибута и (3) дужини временске серије. За једну излазну вредност, улазна временска серија има следећу форму:

$$\begin{bmatrix} t-(tl-1)\Delta t \alpha & t-(tl-1)\Delta t \lambda & t-tl\Delta t \sigma_m & \frac{\partial t-tl\Delta t \sigma_m}{\partial t-tl\Delta t e} \\ t-(tl-2)\Delta t \alpha & t-(tl-2)\Delta t \lambda & t-(tl-1)\Delta t \sigma_m & \frac{\partial t-(tl-1)\Delta t \sigma_m}{\partial t-(tl-1)\Delta t e} \\ \vdots & \vdots & \vdots & \vdots \\ t \alpha & t \lambda & t-\Delta t \sigma_m & \frac{\partial t-\Delta t \sigma_m}{\partial t-\Delta t e} \\ t+\Delta t \alpha & t+\Delta t \lambda & t \sigma_m & \frac{\partial t \sigma_m}{\partial t e} \end{bmatrix}_{(tl+1) \times 4} \quad (6.1)$$

где α представља мишићну активацију, λ је стреч, σ_m је активни напон, $\frac{\partial \sigma_m}{\partial e}$ је извод напона, $(tl+1)$ је укупна дужина временске серије и $t+\Delta t$ је крај тренутног временског корака, унутар симулације методом коначних елемената. На основу улазног тензора неуронска мрежа предвиђа инкремент напона $t+\Delta t \hat{\Delta} \sigma_m$ и инкремент извода напона $\frac{\partial t+\Delta t \hat{\Delta} \sigma_m}{\partial t+\Delta t e}$. Симболом “ $\hat{\Delta}$ ” је означена предикција.



Слика 6.1 Шематски приказ креирања сурогат модела

За обуку модела, скалирају се улазне вредности на опсег између 0 и 1, омогућавајући једнак утицај улазних атрибута на предикцију неуронске мреже. Такође скалирају се излазне вредности напона и извода напона на опсег између 0 и 1. Зарад добијања инкремената, одузимају се одговарајуће скалиране вредности. Инкременти се додатно скалирају, чиме се добијају међусобно разуђеније вредности, па мрежа може да достигне нумерички прецизније предикције. Након обуке мреже, користе се једноставне формуле за рачунање напона и извода напона:

$${}^{t+\Delta t}\hat{\sigma}_m = {}^t\sigma_m + {}^{t+\Delta t}\Delta\hat{\sigma}_m \quad (6.2)$$

$$\partial {}^{t+\Delta t}\hat{\sigma}_m / \partial {}^{t+\Delta t}e = \partial {}^t\sigma_m / \partial {}^te + \partial {}^{t+\Delta t}\Delta\hat{\sigma}_m / \partial {}^{t+\Delta t}e \quad (6.3)$$

Након препроцесирања, креирања и обуке модела, покреће се *Mexie*, користећи исте улазне фајлове, који су претходно коришћени за прикупљање података, али овог пута се користи сурогат модел, уместо оригиналног Хакслијевог модела, за добијање напона и извода напона на нивоу интеграционе тачке. Уколико се добије задовољавајућа сличност између резултата нумеричких експеримената са сурогат и оригиналним моделом, креирају се нови насумични експерименти. У супротном, уколико није достигнута задовољавајућа сличност, враћамо се назад на креирање и обуку модела. Након генерисања нових насумичних експеримената, врши се додатно тестирање сурогат модела, и уколико се добије задовољавајуће поклапање са оригиналним моделом, процес се зауставља. Са друге стране, уколико сурогат не прође додатне тестове, понавља се цео процес од почетка, и додају се нови подаци за креирање сурогат модела. За креирање сурогат модела, генерисано је укупно 160 нумеричких експеримената који су коришћени за обуку и валидацију модела: 45 експеримената са изотоничном контракцијом, 20 експеримената са брзим ослобађањем, 40 експеримената са задатим силама, и 55 експеримената са задатим померањима. Сваки четврти

експеримент је коришћен за валидацију модела, док су остали коришћени за обуку. Генерисано је 8 додатних експеримента, по 2 експеримента сваког типа, за тестирање сурогат модела. Оптимални хиперпараметри неуронских мрежа су пронађени методом пробе и грешке, пратећи процедуру на слици 6.1. Главни критеријум за тестирање квалитета сурогат модела је степен корелације између напона добијених оригиналним и сурогат моделом у току анализе методом коначних елемената. Секундарне метрике за контролу квалитета сурогат модела су време извршавања и потрошња меморије.

6.2 Креирање сурогат модела заснованог на неуронским мрежама подржаним Хакслијевом једначином

Приликом креирања сурогат модела заснованог на вештачким неуронским мрежама подржаним физичким законима могу се посматрати два различита случаја: изометријска и изотонична контракција. Код изометријске контракције брзина клизања филамената је једнака нули и мишићна активација је константна, па је Хакслијева једначина једноставнија за решавање:

$$\frac{\partial n(x,t)}{\partial t} = [1 - n(x,t)]f(x,1) - n(x,t)g(x), \forall x \in \Omega \quad (6.4)$$

при чему су функције f и g стопе закачињања и откачињања миозинске главе за актински сајт. Из једначине (6.4) се види да неуронска мрежа треба да апроксимира вероватноћу закачињања миозинске главе за актински сајт $n(x,t)$ која зависи од позиције миозинске главе x и времена t . Пошто основна формулација неуронских мрежа подржаним физичким законима не захтева обележене податке, могу се насумично генерисати вредности за x и t , из интервала од интереса, и користити за обуку вишеслојног перцептрона при чему је функција губитка дефинисана тако да се минимизује резидуална вредност једначине (6.4) као и резидуална вредност почетног услова $n(x,0) = 0$. Након успешне обуке добија се сурогат модел који решава изометријски случај. Код изотоничне контракције брзина клизања филамената v је различита од нуле и једначина коју треба решити је:

$$\frac{\partial n(x,t)}{\partial t} - v \frac{\partial n(x,t)}{\partial x} = \mathcal{N}(n(x,t), x), \forall x \in \Omega \quad (6.5)$$

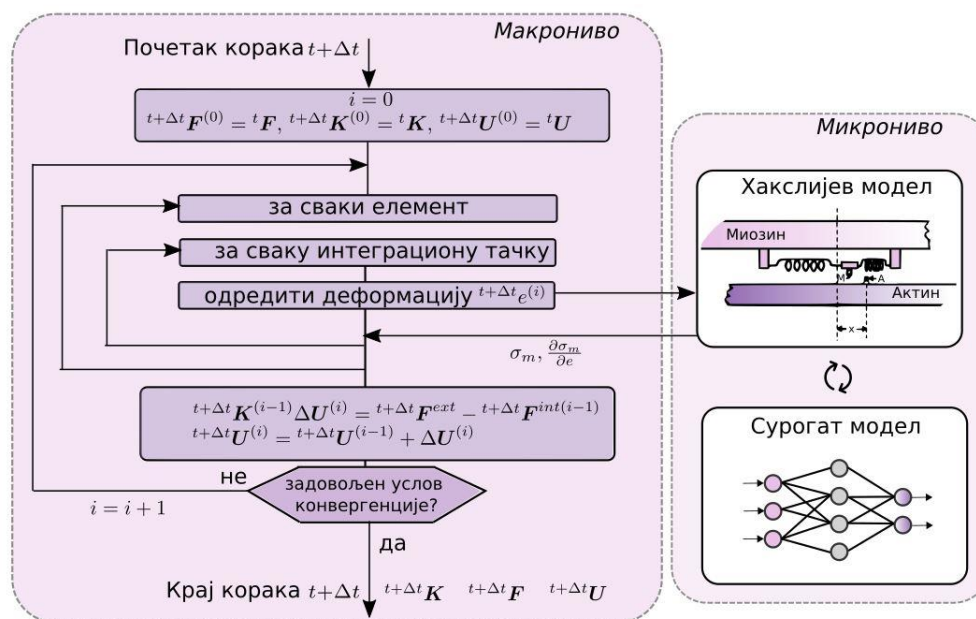
где је \mathcal{N} брзина промене стања попречних мостова која се дефинише као:

$$\mathcal{N}(n(x,t), x) = \begin{cases} [\omega(\lambda) - n(x,t)]f(x,a) - n(x,t)g(x) & , \omega(\lambda) > n(x,t) \\ -n(x,t)g(x) & , \omega(\lambda) \leq n(x,t) \end{cases} \quad (6.6)$$

где a представља мишићну активацију, а $\omega(\lambda)$ представља корекциони фактор преклапања, који се може израчунати као однос смањене дужине преклапања и почетне дужине преклапања. Треба размотрити и доступност сајтова на актину, са којима је могуће успоставити везу у току контракције. Када је издужење велико, смањују се преклапања актинских и миозинских филамената, па је мањи и број актинских сајтова за које се миозинске главе могу везати, што доводи до смањене генерисане силе. Када је скраћење мишића превелико, суседне актинске могу да се преклапају и међусобно ометају. Да би у обзир били узети ови фактори, корекција се дефинише у форми део по

7 Примена сурогат модела у анализи методом коначних елемената

Подсећања ради, у вишескалној симулацији у којој се метод коначних елемената користи на макронивоу, треба заменити Хакслијев модел рачунски ефикаснијим сурогат моделом. Шематски приказ методе коначних елемената са Хакслијевим моделом на макронивоу је дат на слици 7.1. У овом поглављу су приказане процедуре за интеграцију сурогат модела у софтверски оквир за анализу методом коначних елемената. Процедура за интеграцију **сурогат модела, заснованог на неуронским мрежама за рад са временским серијама**, у оквир за анализу методом коначних елемената се састоји од иницијализације, постављања улазних вредности за неуронску мрежу, добављања вредности за напон и извод напона, и ажурирања улаза у мрежу на крају сваког корака методе коначних елемената [70]. Целокупна процедура је представљена на слици 7.2.

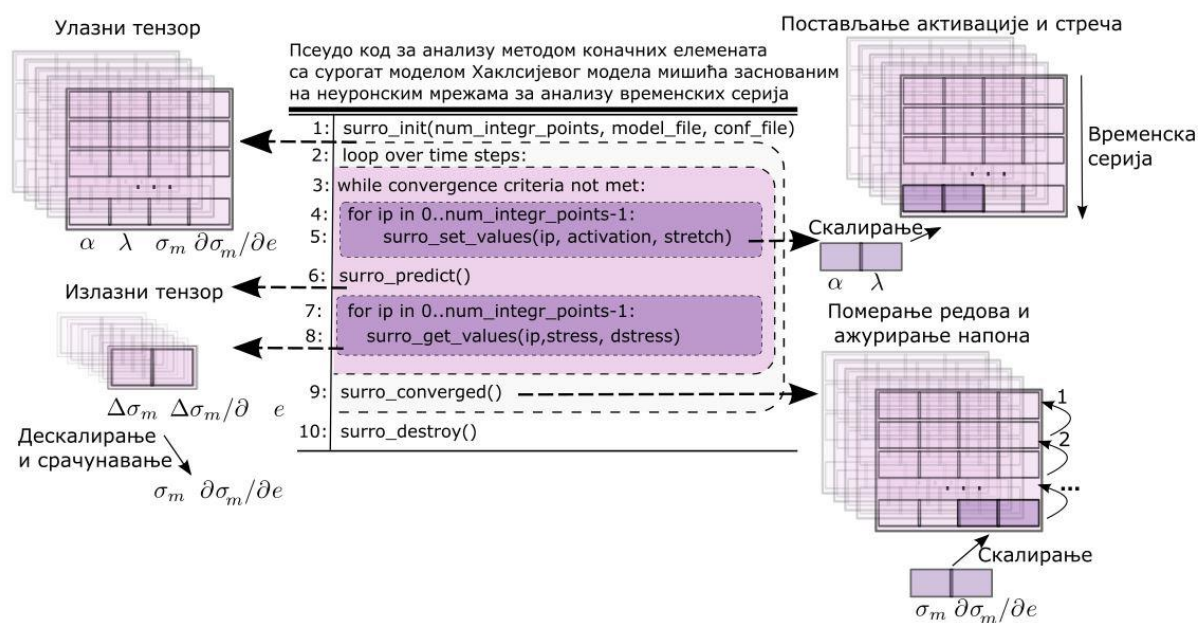


Слика 7.1 Шематски приказ методе коначних елемената на макронивоу и Хакслијевог односно сурогат модела на макронивоу

У току иницијализације (функција *surro_init* на слици 7.2), формира се улазни тензор за неуронску мрежу, учитава се архитектура мреже заједно са тежинским коефицијентима, учитавају се параметри за временску серију, параметри за скалирање и дескалирање података из конфигурационих фајлова. Величина улазног тензора зависи од дужине временске серије, броја улазних атрибута и броја интеграционих тачака у моделу. Иницијалне вредности за стреч су постављене на 1, док су остале вредности у улазном тензору 0. У току сваког корака и сваке итерације нумеричке симулације, за сваку интеграциону тачку (*ip*), вредности за активацију и стреч се предају функцији *surro_set_values*. Ове вредности се скалирају и чувају на одговарајућим локацијама у улазном тензору како је приказано на слици 7.2. Када су све вредности постављене, позивом процедуре *surro_predict* неуронска мрежа предвиђа вредности за инкремент напона и инкремент извода напона. Предвиђање излазних вредности за сваку интеграциону тачку понаособ није довољно рачунски ефикасно. Ради постизања

задовољавајућег убрзања извршавања симулације, треба предвидети излазне вредности за све интеграционе тачке у једном пролазу кроз неуронску мрежу. Након предикције, за потребе коришћења излазних вредности у анализи методом коначних елемената, креирана је процедура *surro_get_values*, која дескалира предикције и на основу њих рачуна напоне (*stress*) и изводе напона (*dstress*), за специфицирану интеграциону тачку (*ip*). На крају корака макронивоа, односно методе коначних елемената, подаци у улазном тензору се померају уназад, остављајући последњи ред за наредни корак нумеричке симулације. Последњи ред тензора представља улазе за тренутни корак симулације, док претходни редови прате претходне кораке симулације, тиме формирајући секвенцу података тј. временску серију за неуронску мрежу.

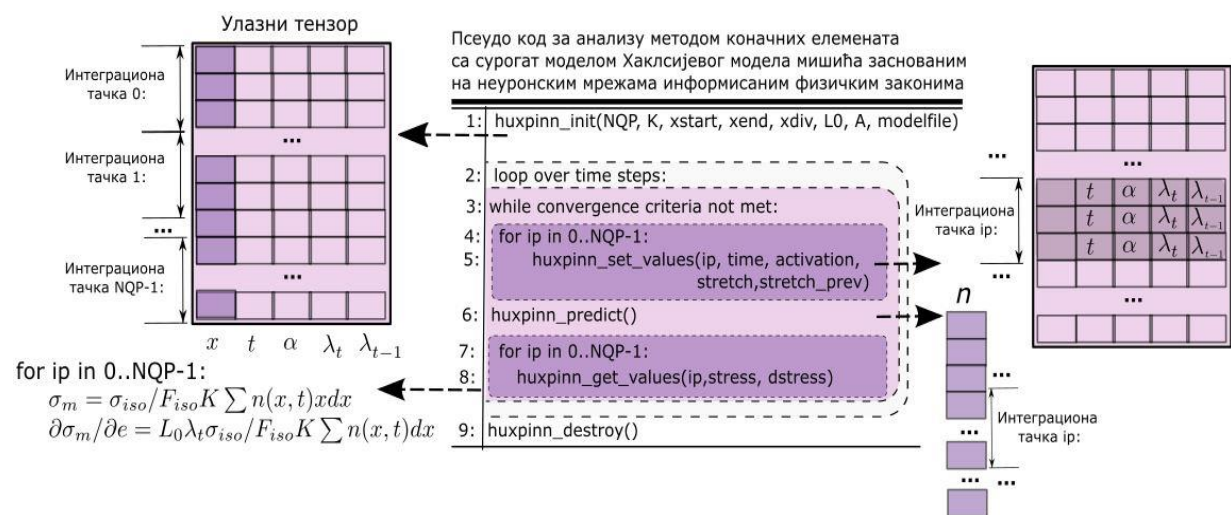
Интеграција **сурогат модела заснованог на неуронским мрежама подржаним физичким законима** у оквир за анализу методом коначних елемената се састоји од иницијализације, постављања улазних вредности за неуронску мрежу и добављања вредности за напон и извод напона. Псеудо код за анализу методом коначних елемената са сурогат моделом заснованим на неуронским мрежама подржаним физичким законима је приказан на слици 7.3.



Слика 7.2 Шематски приказ интеграције сурогат модела заснованог на неуронским мрежама за рад са временским серијама у софтверски оквир за анализу методом коначних елемената

У току иницијализације учитава се неуронска мрежа из фајла *modelfile* и постављају се параметри који се односе на простор у ком се посматра миозинска глава (*xstart* – почетна координата, *xend* – крајња координата, *xdiv* – подела по “*x*”) и остали параметри потребни за касније срачунавање напона и извода напона (*L0* – почетна дужина саркомере, *A* – површина попречног пресека, *K* – крутост попречних мостова). У току иницијализације се такође резервише простор за улазни тензор за неуронску мрежу, чија величина зависи од броја атрибута, од параметра *xdiv* и броја интеграционих, односно Гаусових тачака *NQP* у моделу. Улазни атрибути неуронске

мреже су посматрана позиција миозинске главе x , временски тренутак t , тренутна вредност за активацију α , тренутни стреч λ_t и стреч из претходног временског корака λ_{t-1} . Како је број улазних атрибута пет, величина улазног тензора ће бити $(NQP * xdiv) \times 5$. За сваку интеграциону тачку у моделу, улазни тензор се попуњава вредностима за посматране позиције “ x ” почевши од вредности $xstart$ до вредности $xend$, при чему је укупан број вредности за “ x ” по интеграционој тачки $xdiv$. Ове вредности остају непромењене до краја извршавања симулације. Како се анализа методом коначних елемената одвија, вредности осталих атрибута у улазном тензору се мењају. Функцијом `huxpinn_set_values(ip, time, activation, stretch, stretch_prev)` се за интеграциону тачку са индексом ip , постављају вредности за активацију, време, тренутни и претходни стреч, унутар дела улазног тензора који се односи на ту интеграциону тачку. Набројане вредности се понављају $xdiv$ пута за сваку интеграциону тачку. Када су постављене све улазне вредности, за све интеграционе тачке модела, позива се функција `huxpinn_predict` којом се улазни тензор пропушта кроз неуронску мрежа и мрежа даје излазне вредности “ n ” које представљају вероватноће закачињања миозинске главе за актински сајт. На основу предвиђених вредности и улазних параметара, могу се срачунати вредности за напон и извод напона унутар сваке интеграционе тачке. Позивом функције `huxpinn_get_values` се узимају вероватноће закачињања миозинске главе за актински сајт које се односе на интеграциону тачку ip и коришћењем формула приказаних на слици 7.3 срачунавају се напон и извод напона. Коришћењем приказаних процедура за интеграцију, сурогат модели су уграђени у програме за анализу методом коначних елемената *Mexie* и *PAK-FIS*.



Слика 7.3 Шематски приказ интеграције сурогат модела заснованог на PINN у софтверски оквир за анализу методом коначних елемената

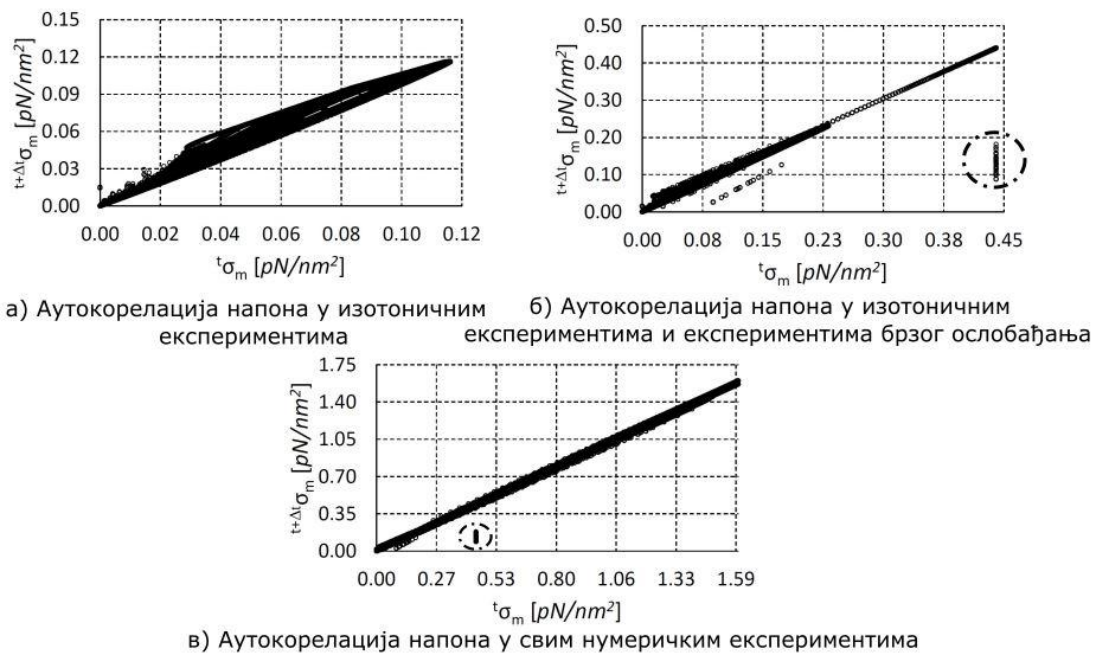
8 Резултати и дискусија

У овом поглављу ће бити представљени резултати сурогат модела добијени у различитим нумеричким експериментима, биће анализирано добијено убрзање, као и потрошња меморије. Одвојено ће бити приказани резултати добијени сурогат моделима заснованим на неуронским мрежама за анализу временских серија и на мрежама подржаним Хакслијевом једначином за мишићну контракцију. На крају поглавља биће приказани модели леве коморе, у којима је сурогат модел коришћен као замена за Хакслијев модел мишића.

8.1 Анализа прикупљених података и механизма обуке неуронске мреже

Аутокорелација напона може да прикаже природу сакупљених података и погодна је за објашњавање зашто предвиђамо инкременте напона уместо директних вредности. На слици 8.1, приказани су напони у тренутку $t+\Delta t$ и у тренутку t . Могу се разликовати три групе података на слици 8.1: (1) подаци из експеримената са изотоничном контракцијом, (2) подаци из експеримената са изотоничном контракцијом и брзим ослобађањем, и (3) сви подаци из нумеричких експеримената коришћених за обуку.

Може се приметити да је аутокорелација између напона ${}^t\sigma_m$ и ${}^{t+\Delta t}\sigma_m$ јака, али када се у скуп за обуку убаце експерименти са брзим ослобађањем појављују се тачке које значајно одступају (заокружене тачке на слици 8.1). У општем случају, напон из тренутка t се повећава или смањује за малу вредност у тренутку $t+\Delta t$. Велика разлика између тренутног и претходног мишићног напона се дешава у току експеримената са брзим ослобађањем, у тренуцима када се сила нагло спушта. Отуда тачке са значајним одступањем на слици 8.1. Уколико бисмо користили неуронске мреже да директно предвиђамо напон ${}^{t+\Delta t}\sigma_m$ мрежа би користила јаку аутокорелацију напона, што би могло да произведе одговарајуће резултате у експериментима са изотоничном контракцијом. Међутим, након додавања експеримената са брзим ослобађањем у скуп за обуку, овај приступ није погодан. Најпре, обука је тежа јер је опсег вредности за напон значајно већи након увођења експеримената са брзим ослобађањем. Велики опсег вредности отежава предвиђање малих и великих вредности напона довољно прецизно, што је посебно уочљиво када се сурогат модел користи унутар симулације за анализу методом коначних елемената. Пошто број тачака које одступају није велики у односу на укупан број тачака, иако би у току обуке, предикције у овим тачкама биле погрешне, укупна вредност функције губитка би била ниска. Уколико тестирамо неуронску мрежу само на подацима, предикције у тренутку спуштања силе би биле погрешне, али све друге предикције, пре и после тренутка ослобађања, би биле добре. Међутим, уколико бисмо покретали симулације за анализу методом коначних елемената и користили сурогат модел на микронивоу уместо Хакслијевог модела, експерименти са брзим ослобађањем би били неуспешни - од тренутка ослобађања па надаље резултати би били погрешни или се симулација не би ни завршила услед дивергенције.



Слика 8.1 Аутокорелација прикупљених вредности за напон у мишићном влакну

Предлог разрешења овог проблема је предвиђање инкремента напона и инкремента извода напона. Предвиђањем инкремената, појачава се важност тренутка у ком се напон значајно мења. Такође, тиме се неуронској мрежи отежава злоупотреба јаке аутокорелације улазног и излазног напона, а омогућава се да заиста научи промене у напонима у односу на улазне атрибуте (активацију, стреч, напон и извод напона). Да бисмо омогућили мрежи да предвиђа мале и велике инкременте довољно прецизно, уведен је фактор скалирања инкремената као хиперпараметар.

Постоји неколико кључних механизма који су довели до успешног обучавања неуронске мреже као сурогат модела мишића:

- (1) предвиђање инкремента напона и извода напона,
- (2) скалирање предвиђених вредности,
- (3) Хуберова функција губитка,
- (4) нормализација градијената.

Раније заустављање обуке омогућава избегавање проблема преучавања. Одсецање градијената је техника која се користи за избегавање експлодирајућих градијената код дубоких неуронских мрежа, а посебно код рекурентних. Хуберова функција губитка одсеца вредности градијената, пошто је извод Хуберове функције константан за грешке изнад задате вредности параметра δ . За нормализовање градијената се користи унапред задат праг. Норме градијената које превазилазе вредност прага се скалирају тако да одговарају прагу, што омогућава стабилнију обуку без наглих промена, мање подложну проблему експлодирајућих градијената. Нормализовање градијената омогућава оптимизационом алгоритму да се понаша боље чак и у пределима функције губитка који су јако неправилни. Без нормализације и одсецања градијената, параметри могу да направе велики спуст и прескоче регион функције губитка са потенцијалним оптималним решењем. Са одсецањем градијената, спуст је ограничен и вредност параметара остаје у добром региону.

8.2 Поређење обучених неуронских мрежа за анализу временских серија

У току креирања сурогат модела заснованих на неуронским мрежама за анализу временских серија, конструисано је пуно различитих мрежа. У овој подсекцији ће бити представљено неколико репрезентативних мрежа, које су показале највећи потенцијал за сурогат моделирање мишића. У табели 8.1, приказани су сумирани хиперпараметри за пет различитих типова неуронских мрежа: (1) *TCN*, (2) *Nested LSTM*, (3) *GRU*, (4) *Nested LSTM-TCN*, and (5) *GRU-TCN*. Хиперпараметри су већином оптимизовани методом покушаја и грешке, пошто није унапред познато која од ових мрежа ће дати најбоље резултате унутар вишескалне анализе методом коначних елемената. *Rectified Adam* се показао као најбољи оптимизациони алгоритам за овај проблем, па је коришћен за све неуронске мреже. Остали заједнички хиперпараметри за све мреже су приказани на дну табеле 8.1. Укупан број тежинских коефицијената је сличан код свих неуронских мрежа, да би биле поређене рачунске перформансе мрежа што приближније сложености. Просечни коефицијенти корелације између правих и предвиђених напона за пет наведених мрежа су дати у табели 8.2. Приказани су коефицијенти корелације добијени на тренинг, валидационом и тестном скупу података, заједно са коефицијентима корелације добијеним у тренинг, валидационим и тестним нумеричким симулацијама. Квалитет сурогат модела се огледа у веома великим просечним вредностима коефицијента корелације између правих и предвиђених вредности као и у малој стандардној девијацији, која показује да сурогат даје предикције сличне прецизности у свим изведеним нумеричким експериментима. Добијени коефицијенти корелације су увек нижи у нумеричким симулацијама него коефицијенти добијени приликом предвиђања на скуповима података. За предикције, на скуповима података, увек се користе улазни подаци који одговарају оригиналном Хакслијевом моделом. Насупрот томе, у току симулације настају разлике у улазном тензору, у односу на прикупљене податке, под утицајем предикција напона и извода напона које даје сам сурогат модел. У општем случају, висок коефицијент корелације добијен на подацима, ће резултовати високим коефицијентом корелације у нумеричким симулацијама, под условом да је генерализација довољно добра. Степен генерализације се најбоље уочава на тестном скупу и тестним нумеричким симулацијама. Уколико мрежа даје добре резултате на тестном скупу, можемо да кажемо да је генерализација довољно добра. Мала стандардна девијација коефицијената корелације у различитим нумеричким експериментима је такође добар индикатор генерализације. Најлошији резултати на тестном скупу су постигнути *TCN* мрежом, што показује да конволуционе неуронске мреже генерализују лошије од рекурентних мрежа, када је анализа секвенцијалних података у питању. Највиши степени корелације на тренинг, валидационом и тестном скупу су постигнути *GRU* неуронском мрежом. Са *GRU* мрежом, достигнути су и најпрецизнији резултати унутар нумеричких симулација. Такође се може приметити да *GRU* неуронска мрежа има механизме за ажурирање и ресетовање који су слични као операције у току анализе методом коначних елемената, чиме се може објаснити зашто ова мрежа даје најбоље перформансе у оваквој вишескалној симулацији. Даље, са *GRU* мрежом добијена је најмања стандардна девијација у већини случајева. Интересантно је да комбиновање *Nested LSTM* и *TCN* мреже даје боље резултате него *Nested LSTM* и *TCN* појединачно. *Nested LSTM* даје боље резултате на тестном скупу података од *TCN*. Са друге стране, *TCN* даје боље резултате на тренинг скупу. Комбиновањем ових мрежа користимо предности обеју мрежа *Nested LSTM* и *TCN*. Међутим, такав ефекат није постигнут комбиновањем *GRU* и *TCN*, већ је сама *GRU* мрежа дала боље резултате. Комбиновањем *GRU* и *TCN* добијају се бољи резултати него комбиновањем *Nested LSTM* и *TCN*, али, свеукупно, *GRU* мрежа показује највећи потенцијал за сурогат моделирање мишића.

Табела 8.1 Хиперпараметри одабраних неуронских мрежа

Тип неуронске мреже (енгл.)	Број тежина	Скривени слојеви
<i>TCN</i>	928,706	11 конволуционих слојева \times 192 филтера дилатације = [1,2,4,8,16], величина кернела = 4
<i>Nested LSTM</i>	992,002	1 угњеждени слој, дубина = 8, 128 неурона по дубини
<i>GRU</i>	992,770	[1. 4. и 5.] GRU слој \times 128 неурона, [2. и 3.] GRU слој \times 256 неурона
<i>Nested LSTM-TCN</i>	991,874	1 угњеждени слој, дубина 6, 128 неурона по дубини, 7 конволуционих слојева \times 128 филтера, дилатације = [1,2,4], величина кернела = 4
<i>GRU-TCN</i>	974,978	[1. и 4.] GRU слој \times 64 неурона, [2. и 3.] GRU слој \times 256 неурона, 7 конволуционих слојева \times 128 филтера, дилатације = [1,2,4], величина кернела = 4

Заједнички хиперпараметри: Rectified Adam је коришћен за оптимизацију свих неуронских мрежа са иницијалном стопом учења 10^{-3} , $\beta_1 = 0.99$, $\beta_2 = 0.9999$, clip norm = 10^{-4} . Хуберова функција губитка је минимизована са параметром $\delta = 58 \times 10^{-6}$. Укупан број епоха је постављен на максимално 50000 са величином бача 16384, и раним заустављањем чији је *patience* параметар постављен на 500 епоха.

Табела 8.2 Просечна вредност и стандардна девијација коефицијената корелације између правих и предвиђених вредности напона

Тип неуронске мреже (енгл.)	Просечна вредност коефицијената корелације између правих и предвиђених вредности напона					
	Подаци			Нумеричке симулације		
	тренинг	валидација	тест	тренинг	валидација	тест
<i>TCN</i>	0.9 ⁵ 45	0.9 ⁴ 89	0.9 ³ 87	0.961	0.967	0.634
<i>Nested LSTM</i>	0.9 ³ 79	0.9 ³ 81	0.9 ⁴ 30	0.543	0.632	0.216
<i>GRU</i>	0.9 ⁶ 72	0.9 ⁶ 40	0.9 ⁵ 18	0.9 ³ 77	0.9 ³ 65	0.989
<i>Nested LSTM-TCN</i>	0.9 ⁵ 51	0.9 ⁵ 56	0.9 ⁴ 71	0.981	0.981	0.812
<i>GRU-TCN</i>	0.9 ⁶ 72	0.9 ⁵ 84	0.9 ⁴ 59	0.9 ³ 24	0.998	0.965

Белешка: Нотацијом 9^x је означено да се број 9 понавља x пута.

Тип неуронске мреже (енгл.)	Стандардна дефијација коефицијената корелације између правих и предвиђених вредности напона					
	Подаци			Нумеричке симулације		
	тренинг	валидација	тест	тренинг	валидација	тест
<i>TCN</i>	3.31×10^{-5}	3.78×10^{-5}	2.95×10^{-4}	1.20×10^{-1}	9.29×10^{-2}	5.19×10^{-1}
<i>Nested LSTM</i>	5.18×10^{-4}	4.08×10^{-4}	3.89×10^{-5}	4.03×10^{-1}	3.47×10^{-1}	4.04×10^{-1}
<i>GRU</i>	9.60×10^{-7}	1.38×10^{-6}	1.80×10^{-5}	1.03×10^{-3}	6.83×10^{-4}	2.35×10^{-2}
<i>Nested LSTM-TCN</i>	3.03×10^{-5}	1.25×10^{-5}	5.01×10^{-5}	4.97×10^{-2}	5.28×10^{-2}	3.57×10^{-1}
<i>GRU-TCN</i>	8.70×10^{-7}	4.05×10^{-6}	9.09×10^{-5}	3.04×10^{-3}	4.52×10^{-3}	6.92×10^{-2}

Табела 8.3 Просечна вредност и стандардна девијација коефицијената корелације између правих и предвиђених вредности извода напона

Тип неуронске мреже (енгл.)	Просечна вредност коефицијената корелације између правих и предвиђених вредности извода напона					
	Подаци			Нумеричке симулације		
	тренинг	валидација	тест	тренинг	валидација	тест
<i>TCN</i>	0.9 ⁷ 50	0.9 ⁶ 20	0.9 ⁶ 73	0.976	0.976	0.691
<i>Nested LSTM</i>	0.9 ⁵ 78	0.9 ⁵ 51	0.9 ⁵ 87	0.762	0.823	0.533
<i>GRU</i>	0.9⁷80	0.9⁷80	0.9⁶84	0.9³84	0.9³70	0.9³75
<i>Nested LSTM-TCN</i>	0.9 ⁷ 40	0.9 ⁶ 80	0.9 ⁴ 70	0.987	0.984	0.845
<i>GRU-TCN</i>	0.9⁷80	0.9 ⁶ 60	0.9 ⁶ 25	0.9 ³ 46	0.998	0.986

Белешка: Нотацијом 9^x је означено да се број 9 понавља x пута.

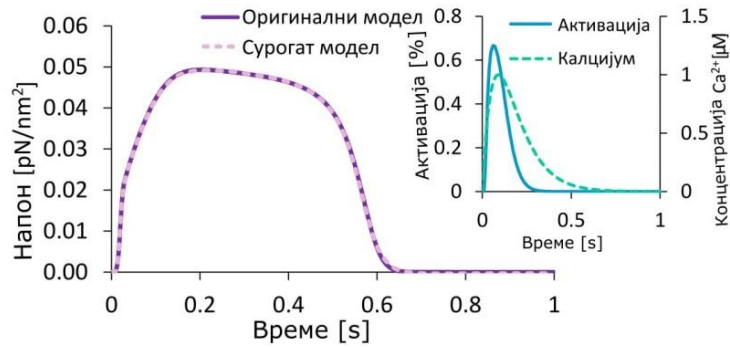
Тип неуронске мреже (енгл.)	Стандардна дефијација коефицијената корелације између правих и предвиђених вредности извода напона					
	Подаци			Нумеричке симулације		
	тренинг	валидација	тест	тренинг	валидација	тест
<i>TCN</i>	1.60 × 10 ⁻⁷	4.20 × 10 ⁻⁶	3.90 × 10 ⁻⁷	1.10 × 10 ⁻¹	7.00 × 10 ⁻²	4.08 × 10 ⁻¹
<i>Nested LSTM</i>	6.35 × 10 ⁻⁶	2.38 × 10 ⁻⁵	1.10 × 10 ⁻⁶	3.57 × 10 ⁻¹	2.41 × 10 ⁻¹	4.63 × 10 ⁻¹
<i>GRU</i>	6.00 × 10 ⁻⁸	3.10 × 10⁻⁷	2.40 × 10⁻⁷	4.61 × 10⁻⁴	6.70 × 10⁻⁴	4.64 × 10⁻⁴
<i>Nested LSTM-TCN</i>	1.60 × 10 ⁻⁷	7.70 × 10 ⁻⁷	4.80 × 10 ⁻⁷	3.60 × 10 ⁻²	3.91 × 10 ⁻²	1.75 × 10 ⁻¹
<i>GRU-TCN</i>	4.00 × 10⁻⁸	1.47 × 10 ⁻⁶	1.43 × 10 ⁻⁶	2.84 × 10 ⁻³	6.83 × 10 ⁻³	2.62 × 10 ⁻²

Може се приметити да коефицијенти корелације опадају у тестним експериментима, код свих неуронских мрежа у табелама 8.2 и 8.3, што указује на то да генерализација може бити даље побољшана. Најмањи пад коефицијената корелације у тестним експериментима се јавља код *GRU* мреже, што додатно показује њен потенцијал за генерализацију. Просечни степени корелације правих и предвиђених вредности за извод напона, тј. тренутну крутост су дати у табели 8.3, која показује сличне резултате као табела 8.2. За спровођење свих нумеричких експеримената, приказаних у овој секцији, коришћен је *Mexie*.

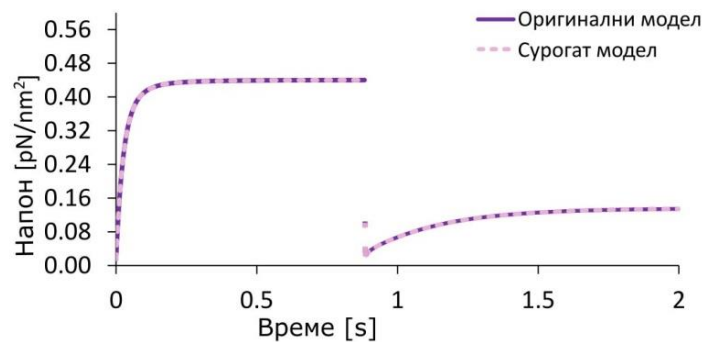
8.3 Дијаграми напона у току времена са *GRU* неуронском мрежом и оригиналним Хакслијевим моделом

У овој подсекцији, упоређени су резултати сурогат модела, базираног на *GRU* мрежи, и оригиналног Хакслијевог модела у различитим типовима нумеричких експеримената. Сlike 8.2-8.13 показују дијаграме напона у времену, добијених у нумеричким симулацијама са оригиналним Хакслијевим моделом и сурогат моделом. Зарад једноставности нису приказани изводи напона, пошто су резултати аналогни. На сликама 8.2-8.5 приказани су неки од случајева који су коришћени приликом обуке мреже. На слици 8.2 приказан је пример изотоничне контрације заједно са задатом концентрацијом калцијума и одговарајућом активационом функцијом. На слици 8.3 приказан је пример експеримента са брзим ослобађањем. На слици 8.4 приказан је пример експеримента са задатом силом, док је на слици 8.5 приказан пример експеримента са задатим померањем. На сликама 8.6-8.9 приказани су неки од случајева за валидацију, док су на сликама 8.10-8.13 приказани неки од случајева за

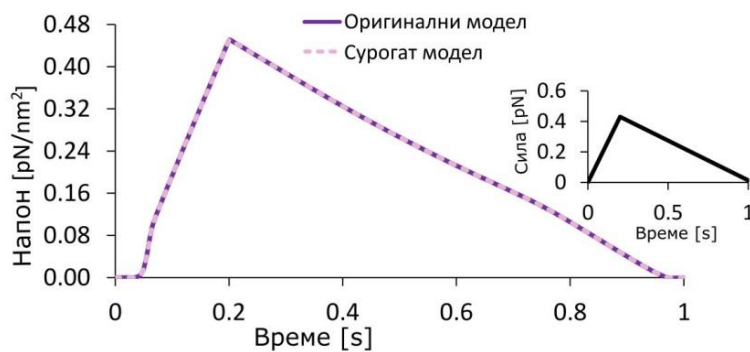
тестирање сурогат модела. У свим овим експериментима сурогат модел је дао врло сличне напоне као оригинални Хакслијев модел мишића. За спровођење нумеричких експеримената, приказаних у овој секцији, коришћен је *Mexie*.



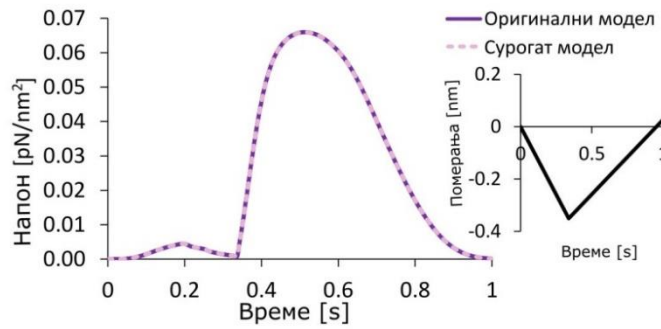
Слика 8.2 Напон добијен у једном од нумеричких експеримената изотоничне контракције из скупа за обуку неуронске мреже



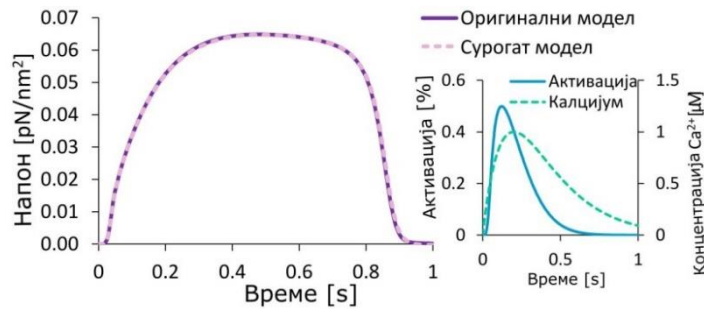
Слика 8.3 Напон добијен у једном од нумеричких експеримената са брзим ослобађањем из скупа за обуку неуронске мреже



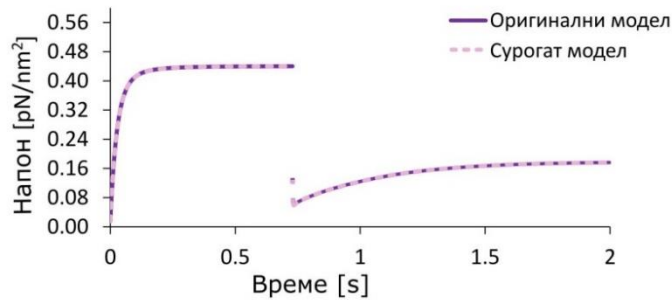
Слика 8.4 Напон добијен у једном од нумеричких експеримената са задатом силом из скупа за обуку неуронске мреже



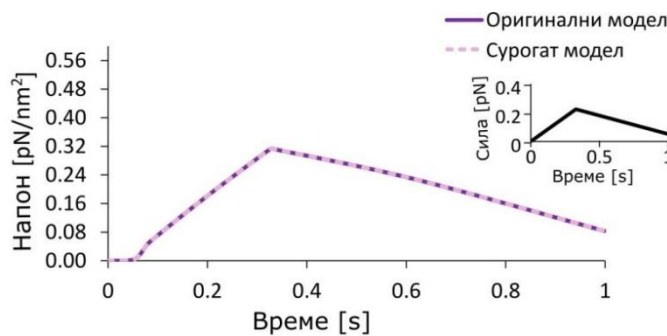
Слика 8.5 Напон добијен у једном од нумеричких експеримената са задатим померањима из скупа за обуку неуронске мреже



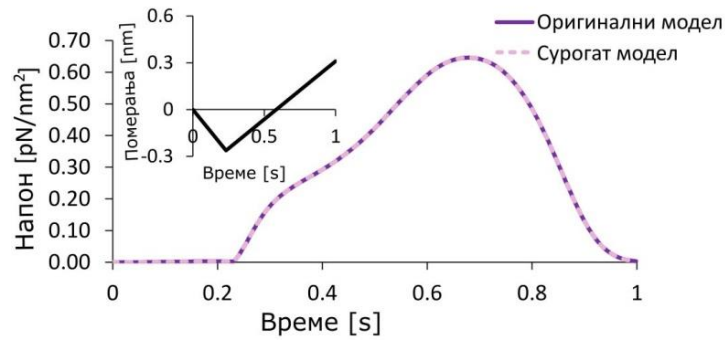
Слика 8.6 Напон добијен у једном од нумеричких експеримената изотоничне контракције из скупа за валидацију неуронске мреже



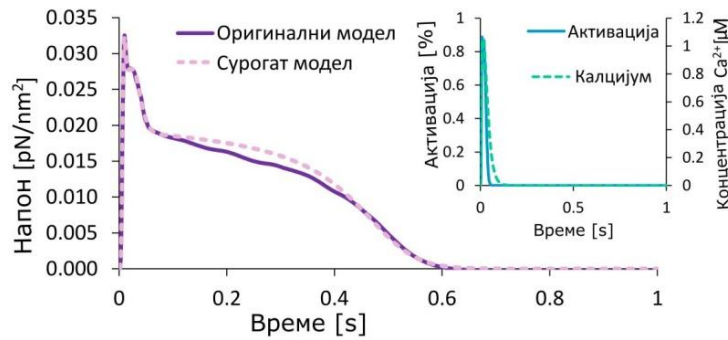
Слика 8.7 Напон добијен у једном од нумеричких експеримената са брзим ослобађањем из скупа за валидацију неуронске мреже



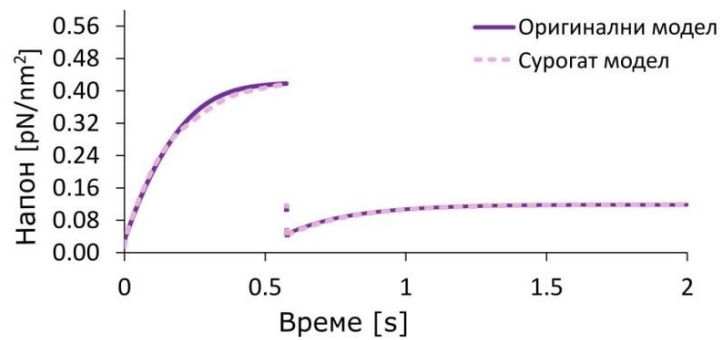
Слика 8.8 Напон добијен у једном од нумеричких експеримената са задатим силама из скупа за валидацију неуронске мреже



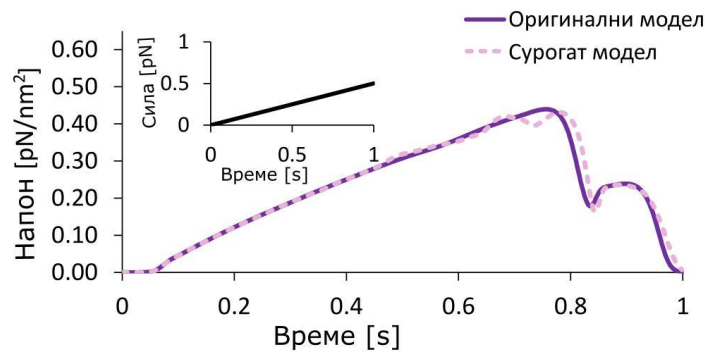
Слика 8.9 Напон добијен у једном од нумеричких експеримената са задатим померањима из скупа за валидацију неуронске мреже



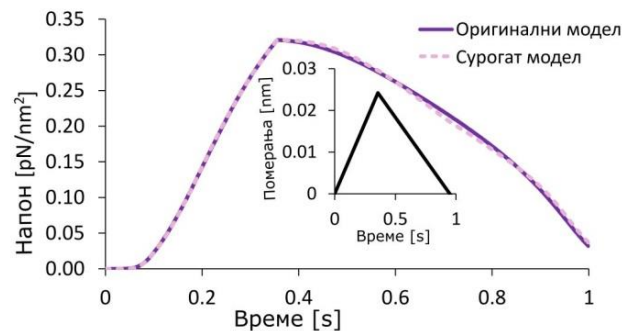
Слика 8.10 Напон добијен у једном од нумеричких експеримената изотоничне контракције из скупа за тестирање неуронске мреже



Слика 8.11 Напон добијен у једном од нумеричких експеримената са брзим ослобађањем из скупа за тестирање неуронске мреже



Слика 8.12 Напон добијен у једном од нумеричких експеримената са задатим силама из скупа за тестирање неуронске мреже

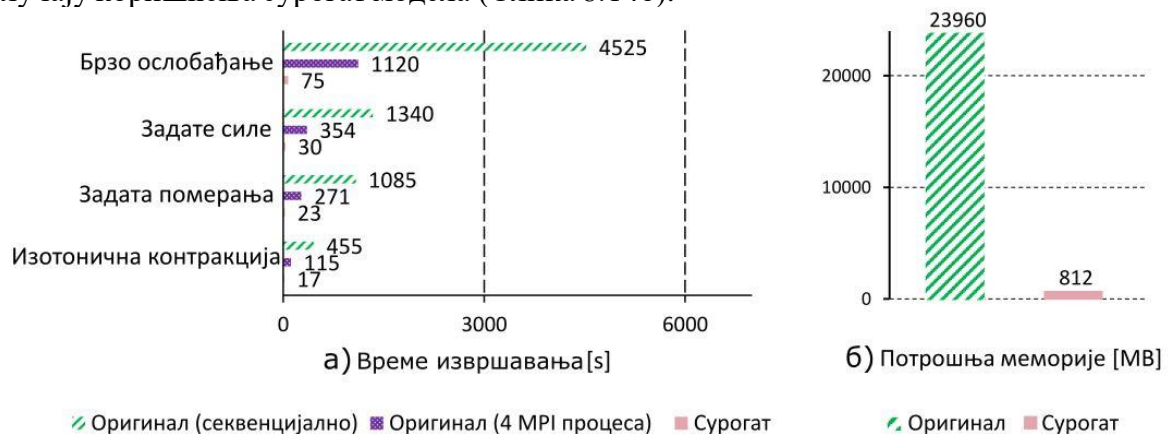


Слика 8.13 Напон добијен у једном од нумеричких експеримената са задатим померањима из скупа за тестирање неуронске мреже

8.4 Убрзање нумеричких симулација добијено суругат моделирањем и потрошња меморије

Главни циљ креираног суругат модела је да буде рачунски ефикаснији него оригинални модел. У овој секцији ће бити анализирано добијено убрзање нумеричких симулација. Добијена убрзања су приказана на слици 8.14а. Приказана су времена извршавања секвенцијалне и паралелне верзије вишескалне симулације са оригиналним Хакслијевим моделом на микронивоу, као и време извршавања симулације са суругат моделом. Паралелизација је извршена на нивоу интеграционе тачке коришћењем *OpenMPI* библиотеке, у којој је имплементиран MPI (енгл. *Message Passing Interface*) стандард за међупроцесну комуникацију. Како сви експерименти садрже 4 интеграционе тачке, покренута су 4 MPI процеса. Суругат модел ради на једном процесору и бржи је од оригиналног модела за ред величине. Прецизније, у поређењу са секвенцијалном верзијом оригиналног модела, суругат модел је 50 пута бржи за експерименте са брзим ослобађањем, задатом силом, и задатим померањима и око 25 пута бржи за експерименте са изотоничном контракцијом. Постигнуто убрзање чини суругат модел употребљивијим у симулацијама са већим бројем коначних елемената. Потрошња меморије суругат модела зависи од величине неуронске мреже, као и величина улазних и излазних тензора, док потрошња меморије оригиналног модела највише зависи од величина низова за чување позиција миозинске главе и вероватноћа формирања попречних мостова на микронивоу. Улазни тензори су релативно мали јер је дужина меморијске серије 11, а број улазних атрибута 4. Насупрот томе низови код оригиналног модела садрже хиљаде бројева са дуплом прецизношћу. У нашем случају, за симулацију модела са 4000 интеграционих тачака, потребно је 23960 MB меморије у

случају коришћења оригиналног Хакслијевог материјалног модела, и само 812 МВ у случају коришћења сурогат модела (Слика 8.14б).

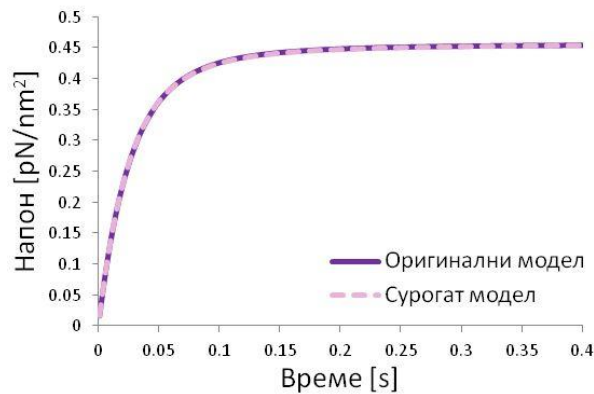


Слика 8.14 Време извршавања симулација и потрошња меморије са оригиналним и сурогат моделом

8.5 Неуронске мреже подржане Хакслијевом једначином за мишићну контракцију

У овој секцији биће дати резултати постигнути неуронским мрежама подржаним Хакслијевом једначином за мишићну контракцију. Ради тестирања способности ових неуронских мрежа да решавају парцијалне диференцијалне једначине, најпре је од интереса изометријски случај, јер је он једноставнији будући да је брзина клизања филамената једнака нули. Код изометријског случаја, вишеслојни перцептрон као улаз узима позицију (x) доступног актинског сајта у односу на равнотежни положај миозинске главе и време (t), а предвиђа вероватноће закачињања миозинских глава за актинске сајтове (n), на основу којих се рачунају напони и тренутна крутост. Конструисана је мрежа са 8 слојева, са по 20 неурона и хиперболичном тангентном активационом функцијом. Генерисан је грид података за 130 еквидистантних вредности за x у опсегу $-20.8 \text{ [nm]} \leq x \leq 62.4 \text{ [nm]}$ и са 500 насумичних вредности за t у опсегу $0 \text{ [s]} \leq t \leq 2.0 \text{ [s]}$. Ове генерисане тачке су коришћене као улазни подаци за обуку мреже. У току обуке, од 30000 епоха, минимизује се резидуал Хакслијеве једначине за мишићну контракцију и резидуал почетног услова, коришћењем Адам оптимизације са стопом учења 10^{-4} и величином *batch*-а 512. Метода неуронских тангентних кернела је коришћена за процену адаптивних тежина циљних функција, који су срачунавани на сваких 300 епоха, чиме се балансира разлика између броја колокационих тачака за минимизовање резидуала парцијалне диференцијалне једначине, и броја тачака коришћених за минимизовање резидуала почетног услова.

Након завршене обуке неуронске мреже, она се може користити као замена за метод карактеристика. На слици 8.15 су дати напони добијени оригиналним моделом, где је Хакслијева једначина решавана методом карактеристика, и сурогат моделом, где неуронска мрежа даје апроксимацију решења Хакслијеве једначине. На слици 8.15 се може приметити да је сличност између напона велика. Степен корелације између напона добијеног оригиналним и сурогат моделом износи 0.9999892. Ово показује да неуронска мрежа решава изометријски случај Хакслијеве једначине довољно добро.

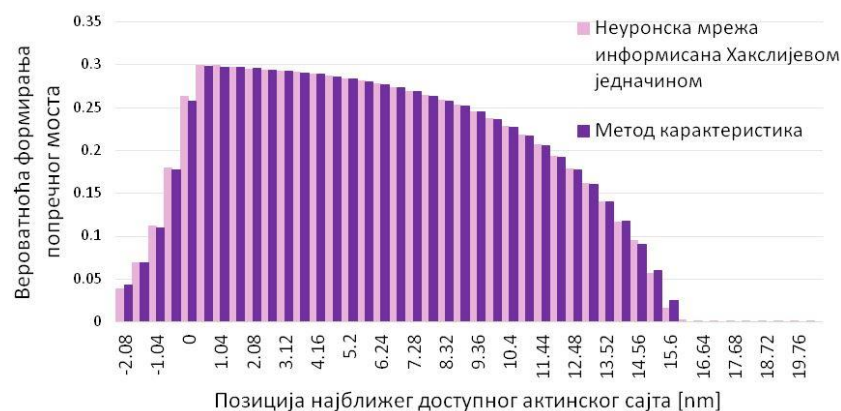


Слика 8.15 Напони у току изометријске контракције добијени оригиналним Хакслијевим моделом и сурогат моделом заснованим на PINN

Код изотоничне контракције постоји брзина клизања филамената па је Хакслијева једначина компликованија за решавање. Поред вредности x и t , вишеслојни перцептрон као улазе користи и мишићну активацију α_t , тренутни стреч λ_t и претходни стреч λ_{t-1} . Број скривених слојева и неурона у сваком од скривених слојева је исти као и код изометријског случаја. За обуку неуронске мреже коришћени су подаци прикупљени из 7 нумеричких симулација са изотоничном контракцијом, где је у свакој од симулација коришћена другачија функција активације. Поред минимизовања разлике предвиђених и прикупљених вредности за n минимизују се и резидуали Хакслијеве једначине и њеног почетног услова. За стопу учења је узета вредност 5×10^{-5} , величина *batch*-а је 16384, број епоха је 7000, при чему се на сваких 400 епоха рачунају тежински коефицијенти циљних функција методом неуронских тангентних кернела. Резидуал Хакслијеве једначине је помножен коефицијентом 10^{-3} да би се смањио утицај великих коефицијената у једначини, на обуку мреже и избегао проблем експлодирајућих градијената. Након обуке су поређене вероватноће формирања попречних мостова добијене методом карактеристика и вишеслојним перцептроном подржаним Хакслијевом једначином. На сликама 8.16 и 8.17 су дате дистрибуције вероватноћа формирања попречних мостова у тренутку $t = 0.5$ [s] за случај који је коришћен у току обуке и случај за тестирање. У оба случаја се може приметити да неуронска мрежа даје приближно исте вероватноће као метод карактеристика.

Да бисмо потврдили да је увођење Хакслијеве једначине у функцију губитка заиста помогло неуронској мрежи да генерализује понашање мишића, упоредићемо вишеслојни перцептрон подржан датим физичким законом са мрежом исте архитектуре, за чију обуку су коришћени само подаци, без специфицирања Хакслијеве једначине. Степени корелације између напона добијених оригиналним моделом и сурогат моделима заснованим на вишеслојном перцептронсу су дати у табелама 8.4 и 8.5. У табели 8.4 су дати случајеви коришћени за обуку мреже, док су у табели 8.5 случајеви за тестирање. У датим табелама се може приметити да се коришћењем вишеслојног перцептрона подржаног Хакслијевом једначином (PINN) добијају напони сличнији оригиналном моделу, него са обичним вишеслојним перцептроном (MLP) са или без *dropout* технике. Такође се може приметити да је, у погледу прецизности

результата, вишеслојни перцептрон са *dropout* техником, приближнији PINN мрежи, него MLP без *dropout* технике. На скупу који је коришћен за обуку, највише степене корелације даје PINN, али најмања стандардна девијација степена корелације се добија са вишеслојним перцептроном са *dropout* техником. На основу тога може деловати да MLP са *dropout* боље генерализује, међутим пошто PINN даје најпрецизније предикције и у тестним експериментима може се закључити да ипак PINN боље генерализује. Треба обратити пажњу и на то да су на скупу за обуку, стандардне девијације степена корелације добијене са PINN и са MLP са *dropout* приближне. Додатно у прилог добре генерализације PINN, иде и најмања стандардна девијација на скупу експеримената за тестирање. У табелама 8.4 и 8.5 су такође дате и вредности за степене корелација између извода напона добијених са оригиналним моделом и сурогат моделима, где се примећују слични ефекти као код напона.



Слика 8.16 Дистрибуција вероватноће формирања попречних мостова у тренутку $t = 0.5$ [s] добијена у току једног примера изотоничне контракције из скупа за обуку



Слика 8.17 Дистрибуција вероватноће формирања попречних мостова у тренутку $t = 0.5$ [s] добијена у току једног примера изотоничне контракције из скупа за тестирање

Табела 8.4 Степени корелације између напона добијених оригиналним моделом и сурогат моделима заснованим на вишеслојном перцептрон у нумеричким експериментима коришћеним за обуку

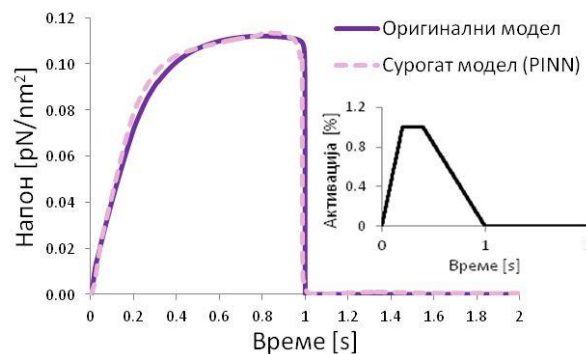
Неуронска мрежа:	PINN		MLP		MLP + dropout	
Редни број нумеричког експеримента	Степен корелације (напон)	Степен корелације (извод напона)	Степен корелације (напон)	Степен корелације (извод напона)	Степен корелације (напон)	Степен корелације (извод напона)
1	0.9929	0.9943	0.9852	0.9872	0.9448	0.9791
2	0.9860	0.9860	0.7782	0.8925	0.9700	0.9833
3	0.9972	0.9958	0.9902	0.9959	0.9286	0.9705
4	0.9343	0.9584	0.1286	0.8354	0.9466	0.9718
5	0.9817	0.9909	0.9964	0.9956	0.9312	0.9742
6	0.9962	0.9893	0.0884	0.8561	0.9495	0.9763
7	0.9978	0.9855	0.1559	0.7753	0.9853	0.9893
Просечна вредност:	0.9837	0.9857	0.5890	0.9054	0.9509	0.9778
Стандардна девијација:	0.0209	0.0117	0.4088	0.0823	0.0189	0.0062

Табела 8.5 Степени корелације између напона добијених оригиналним моделом и сурогат моделима заснованим на вишеслојном перцептрон у нумеричким експериментима коришћеним за тестирање

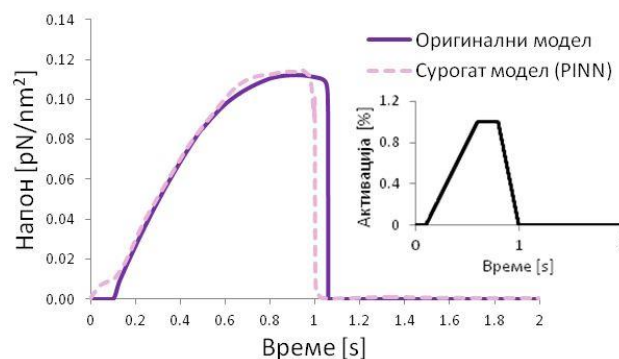
Неуронска мрежа:	PINN		MLP		MLP + dropout	
Редни број нумеричког експеримента	Степен корелације (напон)	Степен корелације (извод напона)	Степен корелације (напон)	Степен корелације (извод напона)	Степен корелације (напон)	Степен корелације (извод напона)
8	0.8861	0.9592	0.6968	0.8795	0.8098	0.9214
9	0.9704	0.9811	0.1854	0.7707	0.9808	0.9859
Просечна вредност:	0.9283	0.9702	0.4411	0.8251	0.8953	0.9536
Стандардна девијација:	0.0421	0.0109	0.2557	0.0544	0.0855	0.0323

На сликама 8.18 и 8.19 су дати дијаграми напона добијених коришћењем PINN и методе карактеристика. Слика 8.18 приказује пример експеримента из скупа за обуку сурогат модела, док слика 8.19 приказује пример експеримента из скупа за тестирање. На сликама 8.20 и 8.21 су дати дијаграми напона добијених коришћењем вишеслојног перцептрона и методе карактеристика у нумеричким експериментима из скупа за обуку и из скупа за тестирање. На слици 8.22 приказано је поређење напона добијених сурогат моделима заснованим на PINN и GRU. Приказане мреже редом међају оригинални Хакслијев модел са $xdiv=45$ и $xdiv=22502$. Обе неуронске мреже дају сличне напоне као оригинални модел, који међају, али GRU мрежом су добијени

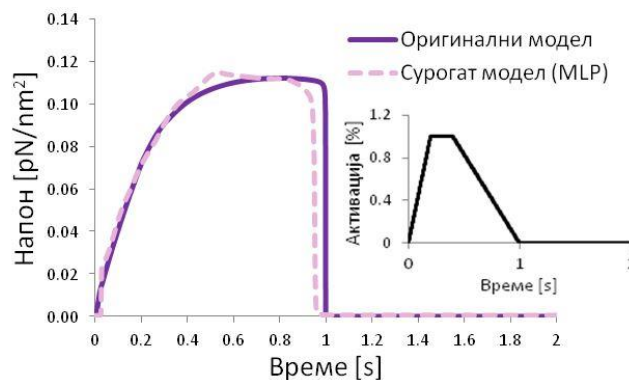
прецизнији резултати. У експерименту приказаном на слици 8.22, добијени степени корелације између оригиналних и предвиђених напона са *PINN* и *GRU* су редом 0.9929 и 0.99998. Посматрањем степена корелације у табелама 8.4, 8.5, А.1 и А.2 (Додатак) може се уочити да је *GRU* мрежа дала прецизније резултате и у осталим нумеричким експериментима.



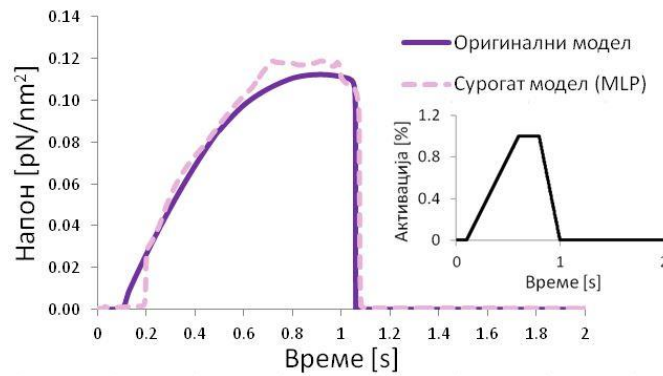
Слика 8.18 Напони добијени оригиналним Хакслијевим моделом и сурогат моделом заснованим на PINN у једном од нумеричких експеримената са изотоничном контракцијом из скупа за обуку



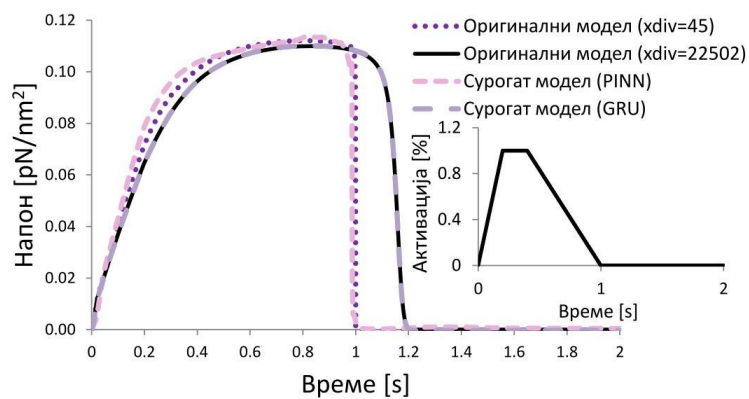
Слика 8.19 Напони добијени оригиналним Хакслијевим моделом и сурогат моделом заснованим на PINN у једном од нумеричких експеримената са изотоничном контракцијом из скупа за тестирање



Слика 8.20 Напони добијени оригиналним Хакслијевим моделом и сурогат моделом заснованим на MLP у једном од нумеричких експеримената са изотоничном контракцијом из скупа за обуку



Слика 8.21 Напони добијени оригиналним Хакслијевим моделом и сурогат моделом заснованим на MLP у једном од нумеричких експеримената са изотоничном контракцијом из скупа за тестирање



Слика 8.22 Поређење напона добијених сурогат моделима заснованим на PINN и GRU

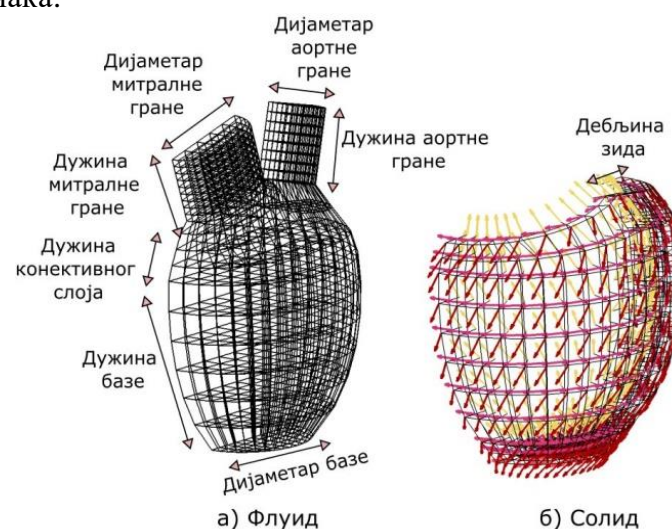
На слици 8.23 је приказано постигнуто убрзање са неуронском мрежом подржаном Хакслијевом једначином у односу на метод карактеристика. Сурогат модел је дуго бржи од оригиналног модела. Треба приметити да је приликом овог поређења и оригинални модел релативно брз. Разлог за брзо извршавање оригиналног модела је коришћење грубе поделе по x . Ова груба подела је коришћена за прикупљање података, па је стога коришћена и на крају за поређење убрзања. Веће убрзање се може добити са финијом поделом по x , али приликом прикупљања података, финија подела по x , може резултовати превеликом количином података, које би било тешко обрадити, и који би знатно успорили обуку неуронске мреже.



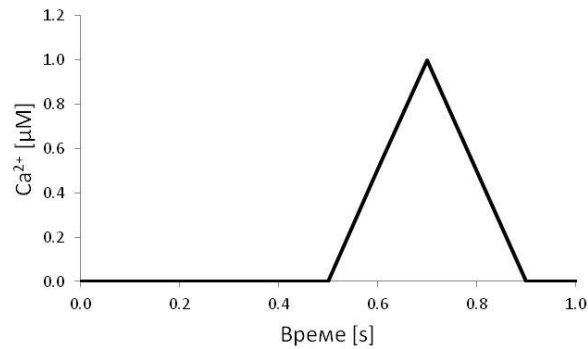
Слика 8.23 Време извршавања нумеричког експеримента изотоничне контракције са оригиналним моделом и PINN сурогат моделом

8.6 Параметарски модел леве коморе

У овој секцији ће бити приказани резултати добијени коришћењем сурогат модела у вишескалној симулацији са већим бројем коначних елемената, на примеру модела леве коморе са параметризованом геометријом. Параметарски модел се може користити за испитивање механичког одзива леве коморе у току срчаног циклуса. Овај модел се састоји од домена флуида и домена солида, као што је приказано на слици 8.24. Моделирана је половина леве коморе са митралним и аортним гранама на врху. Ове гране су повезане, са базним делом коморе, конективном геометријском компонентом (Слика 8.24а). Геометрија домена флуида се генерише задавањем дужина и дијаметара појединачних компоненти. Уведени су и додатни параметри зарад контролисања густине мреже коначних елемената, као што су подела дуж базе коморе, подела дуж грана итд. Домен солида се наставља на домен флуида (Слика 8.24б). За генерисање солид домена, треба задати дебљину зида, као и број слојева дуж зида. Стрелице на слици 8.24б представљају хеликоидална мишићна влакна која су под угловима који варирају од 60° на епикардијуму (жуте стрелице) до -60° на епикардијуму (црвене стрелице). Влакна у средини зида су приказана розе бојом. Коришћена је линеарна промена угла влакна дуж зида леве коморе. Домен солида садржи око 4000 интеграционих тачака.



Слика 8.24 Модел леве коморе: а) домен флуида б) домен солида са мишићним влакнима



Слика 8.25 Задата концентрација калцијума

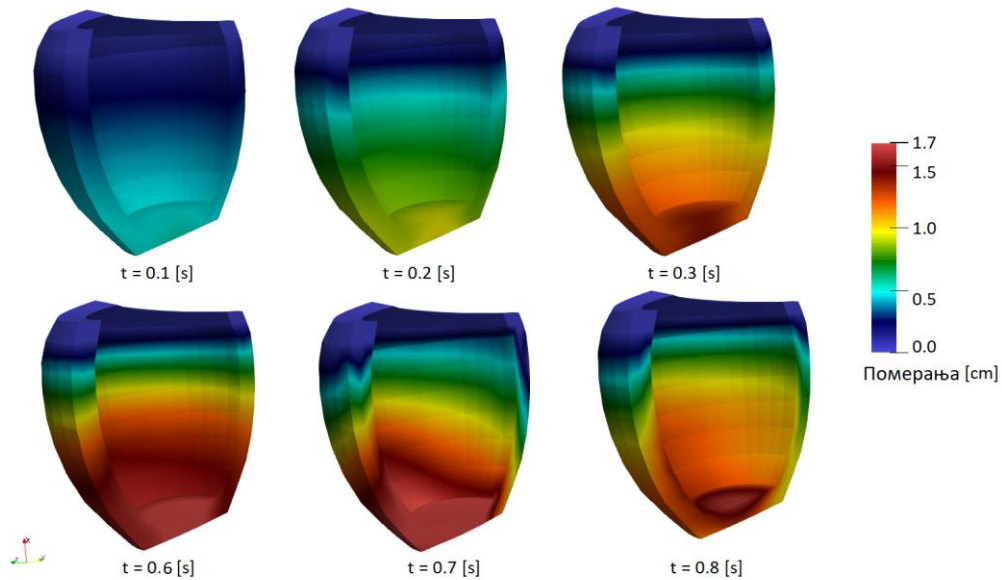
Геометријски параметри модела су дати у табели 8.6, а задата концентрација калцијума је приказана на слици 8.25. Симулација срчаног циклуса креће од почетне конфигурације приказане на слици 8.24, која одговара почетку дијастоле. У току дијастоле, константна брзина од 100 mm/s је задата на митралном улазу, док је аортна грана затворена. На крају периода дијастоле, брзина на митралном залистку пада на нулу, и оба залистка су затворена. Овај временски интервал у току ког су оба залистка затворена одговара изоволуметријској контракцији, и мишићи се активирају у овом периоду. Аортни залистак се отвара на почетку систоле и мишићи се полако деактивирају. Под утицајем сила генерисаних у мишићима крв истиче из леве коморе кроз аортни залистак. За рачунање пасивних напона коришћен је Холзапфелов материјални модел [23-29], док је за рачунање активних напона унутар зида леве коморе коришћен сурогат модел, базиран на GRU мрежи, који је показао највећу сличност са оригиналним Хакслијевим моделом.

Табела 8.6 Геометријски параметри модела леве коморе

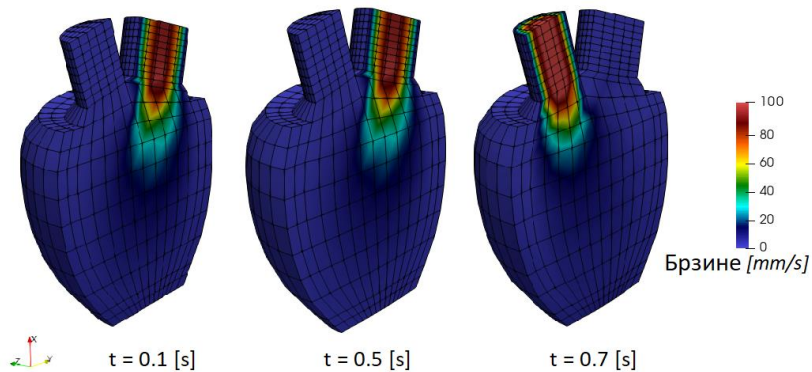
Параметар	Димензије [mm]
Дијаметар базе	57
Дужина базе	45.6
Дужина конективног слоја	19
Дијаметар митралне гране	30.4
Дужина митралне гране	19
Дијаметар аортне гране	17.1
Дужина аортне гране	19
Дебљина зида	7.6

Поља померања у зиду леве коморе на почетку дијастоле ($t=0.1s$, $t=0.2s$, $t=0.3s$), и на почетку и средини систоле ($t=0.6s$, $t=0.7s$, $t=0.8s$) су приказана на слици 8.26. На слици 8.27 су приказана поља брзина на почетку и на средини дијастоле као и на средини систоле. На основу поља померања, може се приметити да мишићи генеришу силу на почетку систоле, контрахујући комору, а затим се полако враћају у првобитни положај услед деактивације. Добијени резултати показују да сурогат модел може бити примењен у симулацијама са већим бројем коначних елемената. Резултати, приказани у овој секцији, су добијени коришћењем *PAK-FIS*-а са урађеним сурогат моделом заснованим на GRU неуронској мрежи. Слични резултати се добијају и са сурогат моделом заснованим на неуронским мрежама подржаним Хакслијевом једначином за

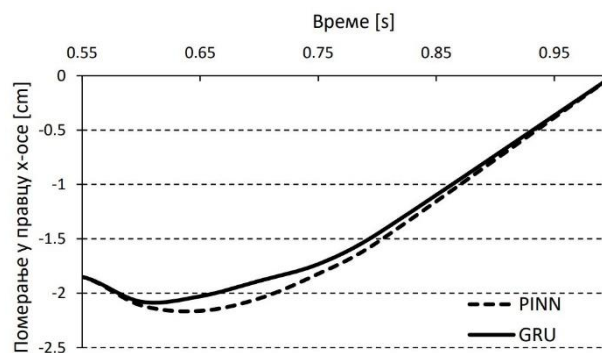
контракцију, што се може видети на слици 8.28 на којој су приказана померања насумично одабраног чвора у правцу x -осе у току систоле.



Слика 8.26 Поље померања на почетку дијастоле ($t=0.1s, t=0.2s, t=0.3s$), и на почетку и средини систоле ($t=0.6s, t=0.7s, t=0.8s$)



Слика 8.27 Поље брзина унутар флуида леве коморе на почетку и средини дијастоле и на средини систоле



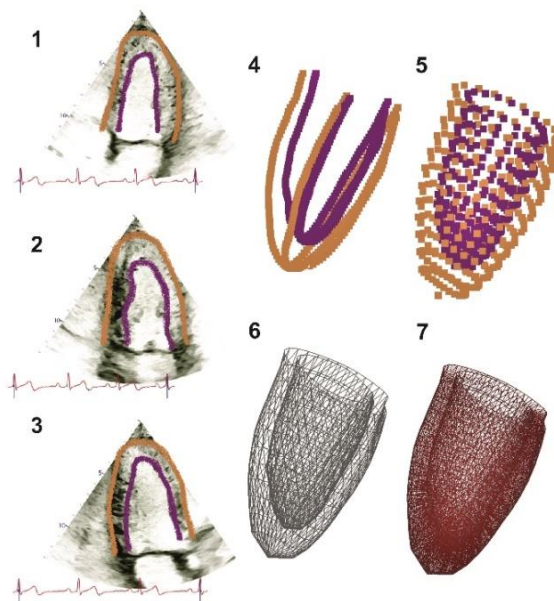
Слика 8.28 Померање насумично одабраног чвора параметарског модела леве коморе у правцу x -осе у току систоле са сурогат моделима заснованим на *PINN* и *GRU*

8.7 Модел леве коморе генерисан на основу ехокардиографских снимака

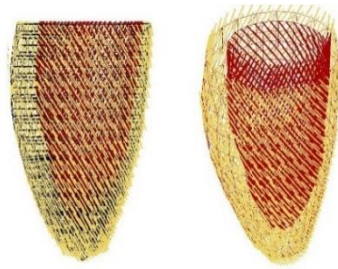
У Универзитетском клиничком центру Србије направљени су ехокардиографски снимци коришћењем Vivid E95 ултразвучног система, при чему је 38-годишњи пацијент лежао у бочном положају. Геометрија леве коморе је генерисана полу-аутоматски са снимака у три равни под угловима 90° , 150° и 210° и померања су задата на унутрашњој површи коришћењем неколико одабраних тренутака, односно конфигурација. Алгоритам за генерисање модела обухвата:

- ❖ Издвајање контура и спецификацију димензија сваке конфигурације.
- ❖ Постављање контура у равнима фиксираним у 3Д простору под угловима од 90° , 150° и 210° .
- ❖ Коришћење сплајн и линеарне интерполације за одређивање положаја тачака на унутрашњој/спољашњој површи зида леве коморе.
- ❖ Генерисање површи на основу тачака.
- ❖ Генерисање мреже коначних елемената за почетну конфигурацију.
- ❖ Генерисање мишићних влакана у почетној конфигурацији.

За почетну конфигурацију се узима почетак дијастоле, а остале конфигурације су одабране тако да прате ток срчаног циклуса до краја. Генерисање мреже коначних елемената и генерисање влакана се врши само за почетну конфигурацију. Мишићна влакна су хеликоидална и постављена су под угловима који варирају од -60° на спољашњем зиду до 60° на унутрашњем зиду, у односу на циркуларни правац коморе. Издвајање контура је урађено мануелно, обележавањем и извлачењем координата са слика. Генерисање модела је илустровано на слици 8.29, где делови 1-3 представљају издвојене контуре у 3 равни, део 4 илуструје издвојене контуре у 3Д простору, део 5 тачке у 3Д простору, 6 површи, а 7 мрежу коначних елемената. Љубичастом бојом су означене контуре на унутрашњем зиду (ендокардијум), а наранџастом на спољашњем зиду коморе (епикардијум). На слици 8.30 су приказани правци генерисаних мишићних влакана.

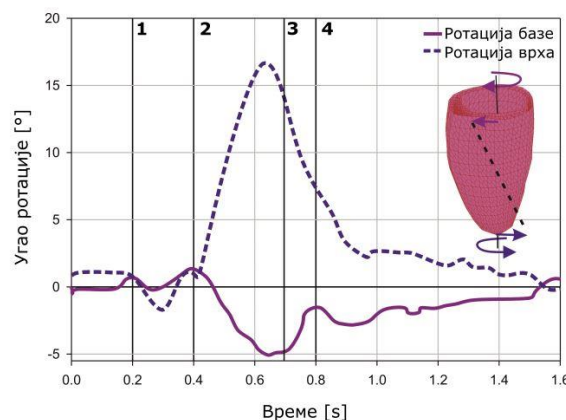


Слика 8.29 Генерисање модела леве коморе на основу ехокардиографских снимака у три равни под угловима 90° , 150° и 210°



Слика 8.30 Мишићна влакна на епикардијуму, приказана жутом бојом, и влакна на ендокардијуму леве коморе, приказана црвеном бојом

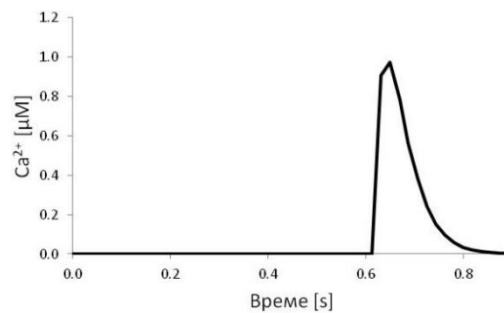
Након генерисања мреже коначних елемената за почетну конфигурацију, за остале одабране конфигурације се генерише само унутрашња површина. На основу генерисаних унутрашњих површина, које представљају кретање ендокардијума у току срчаног циклуса, задата су померања на унутрашњем зиду солида. У обзир се узима и торзија леве коморе, ротирањем генерисаних тачака за одговарајући угао. На слици 8.31 приказани су углови ротације на врху и на базном делу леве коморе. Назначени тренуци 1, 2, 3 и 4 на слици 8.31 се редом односе на изоволуметријску контракцију, систолу, изоволуметријску релаксацију и почетак дијастоле. На базном делу и на врху леве коморе користе се углови из литературе, а за делове који се налазе између врха и базе се врши линеарна интерполација између две вредности на основу положаја односно удаљености од врха/базе модела. У табели 8.7 су дате димензије сваке конфигурације (радијус и висина), као и временски тренутак који треба узети са криве за торзију зарад ротирања конфигурације. Померања у деловима солида који се не налазе на унутрашњем зиду коморе се добијају анализом методом коначних елемената, при чему се за рачунање пасивног дела напона користи експериментални Холзапфелов модел, а за рачунање активног напона у мишићном влакну, користи сурогат модел заснован на *GRU* мрежи или сурогат модел заснован на неуронској мрежи подржаној Хакслијевом једначином за контракцију. Симулација почиње од почетка дијастоле, за коју је узето да траје 0.6 [s], и завршава се крајем систоле за коју је узето да траје 0.3 [s]. На слици 8.32 је приказана задата концентрација калцијума која служи за активацију мишића, а на слици 8.33 су дата поља померања у солиду на почетку дијастоле, на средини и на крају систоле. На слици 8.34 су приказана померања насумично одабраног чвора у току систоле, у правцу *x*-осе са *PINN* и *GRU*, одакле се види да су добијена слична померања са оба сурогат модела.



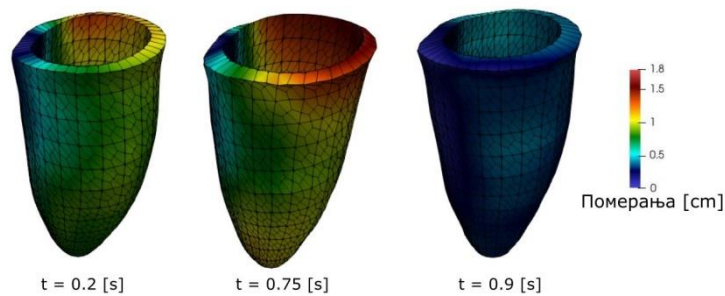
Слика 8.31 Углови ротације у току срчаног циклуса на базном делу и на врху леве коморе

Табела 8.7 Димензије модела леве коморе за 10 одабраних конфигурација са ехокардиографских снимка почевши од почетка дијастоле до краја срчаног циклуса

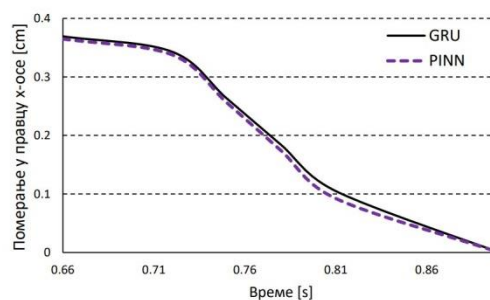
Радијус коморе [cm]	Висина коморе [cm]	Време на кривој за торзију [s]
2.75	9	0.65
2.75	9	0.8
3	9.5	0.95
3.2	9.5	1.1
3.2	9.5	1.25
3.2	9.5	1.4
3.5	9.65	1.55
3.25	9.6	0.2
2.75	9.5	0.4
2.75	9	0.65



Слика 8.32 Задата концентрација калцијума

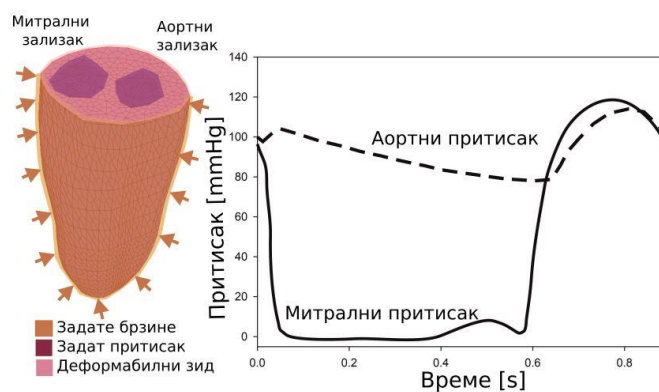


Слика 8.33 Поља померања у солиду леве коморе на почетку дијастоле, средини и крају систоле

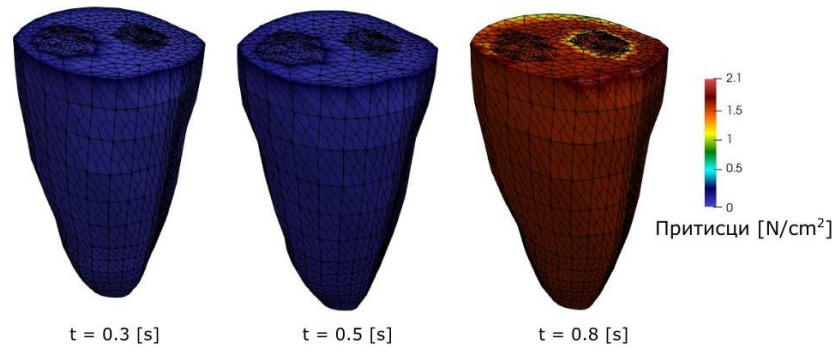


Слика 8.34 Померање насумично одабраног чвора модела леве коморе у правцу x -осе у току систоле са сурогат моделима заснованим на $PINN$ и GRU

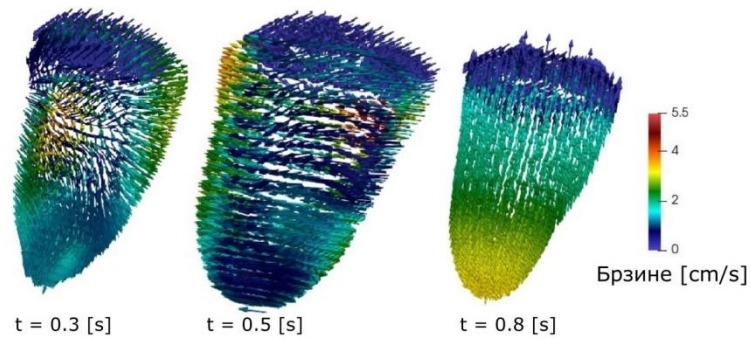
За разлику од параметарског модела, где су солид и флуид компоненте модела чврсто спрегнуте, код модела генерисаног на основу ехокардиографских записа, флуид је моделиран независно од солида. На основу унутрашње површи солида у почетној конфигурацији, генерисан је флуид, а на основу задатих померања на унутрашњем зиду солида, срачунате су и задате брзине на делу флуида, који је, у реалности, у додиру са зидом коморе. На врху модела флуида налази се деформабилни зид са митралним и аортним залистцима. Деформабилни зид се помера тако да су померања у складу са померањем унутрашњег зида. Ово је обезбеђено коришћењем полихармонијских деформација, имплементираних у *libIGL* библиотеци [71], којима се пропагирају задата померања у једном делу модела на друге делове модела тако да се добије гладак облик. На митралном и аортном залистку је задат притисак из литературе. Гранични услови су приказани на слици 8.35. У току дијастоле, аортни залистак је затворен, па су, у овој фази циклуса, задате брзине на аортном залистку једнаке нули. У фази систоле брзине нису задате на аортном залистку. У току дијастоле брзине нису задате на митралном залистку, док је у току систоле, митрални залистак затворен, па су, у овој фази циклуса, задате брзине на митралном залистку једнаке нули. Поља притисака и брзина у флуиду леве коморе на почетку дијастоле, на крају дијастоле и на средини систоле су приказана на сликама 8.36 и 8.37. Притисак је униформан унутар коморе. Брзине у току дијастоле иду ка спољашности коморе, док се на средини систоле крв испумпава, па брзине иду ка залистцима. Средњи притисак у току времена је приказан на слици 8.38. Срачунат притисак се поклапа са уобичајеним притиском код пацијената без поремећаја. Промена притиска у односу на запремину је такође дата на слици 8.38, и приближна је стандардном дијаграму из литературе за пацијенте без поремећаја. Резултати, приказани у овој секцији, су добијени коришћењем *PAK-FIS*-а са урађеним сурогат моделом заснованим на GRU неуронској мрежи за рачунање активних напона и тренутне кривости.



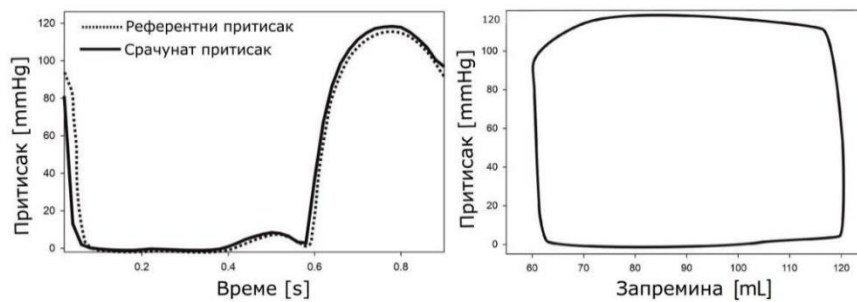
Слика 8.35 Гранични услови за модел флуида леве коморе



Слика 8.36 Поља притисака у флуиду леве коморе на почетку дијастоле, крају дијастоле и средини систоле



Слика 8.37 Поља брзина у флуиду леве коморе на почетку дијастоле, крају дијастоле и средини систоле



Слика 8.38 Притисак у току времена и притисак у односу на запремину

9 Закључна разматрања

Хакслијев модел мишића је биофизички модел заснован на физиологији мишића, прецизнији, али значајно рачунски захтевнији од феноменолошких модела. Рачунски захтеви биофизичких модела, отежавају и у неким случајевима онемогућавају, њихову употребу у вишескалним симулацијама. Да би се овај проблем превазишао, може се користити техника позната као сурогат моделирање, којом се оригинални модел замењује рачунски ефикаснијом апроксимацијом. У овој дисертацији су представљени сурогат модели Хакслијевог модела мишића засновани на вештачким неуронским мрежама. Један од приступа, за креирање сурогат модела, је прикупљање података из рачунарских симулација и обучавање неуронских мрежа тако да оне апроксимирају понашање стварног модела. Други приступ подразумева неуронске мреже подржане физичким законима за приближно решавање парцијалне диференцијалне једначине са почетним и граничним условима.

У овој дисертацији је представљен генератор нумеричких експеримената који креира различите типове експеримената као што су (1) изотонична контракција, (2) брзо ослобађање, (3) задате силе и (4) задата померања. Ови генерисани експерименти се користе као улази у програм за анализу методом коначних елемената са оригиналним Хакслијевим моделом на микронивоу. У току вишескалне симулације методом коначних елемената, у свакој интеграционој тачки, материјални модел на основу стреча, мишићне активације, задатих параметара и интерног стања даје напоне и изводе напона. Улазне вредности у материјални модел, као и његов одзив, се прикупљају у току извршавања симулације. Пошто су мишићи материјал чије понашање је условљено претходном историјом оптерећења, сакупљени подаци су конвертовани у временску серију и коришћени су за обучавање рекурентних и конволуционих неуронских мрежа. Прецизније, за предвиђање напона и извода напона у тренутном временском кораку, као улази у неуронску мрежу се користе: (1) мишићна активација у тренутном и претходним корацима, (2) стреч у тренутном и претходним корацима, (3) напон у претходним корацима и (4) извод напона у претходним корацима. Обучавање неуронских мрежа, такво да се сурогат модел понаша слично унутар методе коначних елемената као оригинални модел, је представљало велики изазов. Прецизност предвиђених напона је морала да буде висока да би симулација методом коначних елемената давала резултате задовољавајуће тачности. Генерализација неуронских мрежа је такође морала да буде довољно добра. У супротном, неуронске мреже би давале лоше резултате у нумеричким експериментима који нису коришћени за прикупљање података и обуку. Уколико је генерализациони јаз неуронске мреже велики, неуронска мрежа може давати лоше резултате и у нумеричким експериментима који су коришћени за обуку, пошто сама мрежа у току нумеричке симулације генерише напоне и изводе који су другачији у односу на оригинални модел, што утиче на будуће предикције у току извршавања симулације. Обучаване су различите рекурентне и конволуционе неуронске мреже за анализу временских серија, као што су (1) *TCN*, (2) *Nested LSTM*, (3) *GRU*, (4) *Nested LSTM-TCN*, и (5) *GRU-TCN*. Да би обука ових мрежа била успешна, неколико механизма машинског учења је примењено, као што су одсецање и нормализација градијената. Један од кључних механизма, који су решили проблеме прецизности и генерализације је спречавање неуронске мреже да злоупотреби високу аутокорељацију напона, предвиђањем инкремената, уместо директних вредности за напон и извод напона. Такође су уведени фактори за скалирање инкремената, у току обуке, чиме је омогућено да мреже дају прецизније предикције. Као најбоља неуронска мрежа за проблем сурогат моделирања мишића, показала се *GRU* неуронска мрежа,

којом је достигнута висока прецизност и довољно мали генерализациони јаз у стотинама различитих нумеричких експеримената.

Други приступ за сурогат моделирање, који је приказан у овој дисертацији, заснован је на неуронским мрежама подржаним физичким законима. Главна иновација је увођење додатне мреже која енкодира парцијалну диференцијалну једначину од интереса, што је у овом случају Хакслијева једначина за мишићну контракцију. Излаз из дубоке неуронске мреже, односно сурогат модела, тада представља апроксимацију решења диференцијалне једначине. Овај излаз се користи као улаз у додатну неуронску мрежу, која енкодира физички закон и којом се добија грешка апроксимације, односно резидуална вредност. Циљ је минимизовати резидуалну вредност, коришћењем техника аутоматске диференцијације и оптимизације, и тиме добити неуронску мрежу која решава једначину са задовољавајућом тачношћу. У овој дисертацији је приказано приближно решавање Хакслијево једначине за изометријску контракцију, коришћењем вишеслојног перцептрона. Показано је да постоји велика сличност између решења добијених методом карактеристика и решења добијених неуронским мрежама подржаним Хакслијевом једначином за изометријску контракцију. У случају изотоничне контракције, поред минимизације резидуала парцијалне диференцијалне једначине и резидуала почетног услова, у циљу приближног решавања Хакслијево једначине за изотоничну контракцију, укључена је и минимизације разлике између предвиђених и прикупљених вредности из нумеричких симулација.

Сурогат модел, заснован на GRU неуронској мрежи, се показао за ред величине бржим од оригиналног модела, што омогућава симулирање понашања већих, реалистичнијих и рачунски захтевнијих модела. Демонстриран је потенцијал овог сурогат модела, на примеру симулирања срчаног циклуса леве коморе, што би било много теже постићи са оригиналним Хакслијевим моделом због сложености микромодела и велике потрошње рачунарског времена и меморије. Приказана су два модела леве коморе, параметарски и модел генерисан на основу ехокардиографских снимака. У оба случаја симулиран је срчани циклус, почевши од почетка дијастоле, па до краја систоле. Лева комора се најпре шири услед дотока крви, а затим се мишићи активирају почетком систоле, комора се контрахује и крв одлази из ње, чиме се завршава циклус. Поред процедура за креирање сурогат модела, у овој дисертацији представљене су и процедуре за интеграцију сурогат модела у софтверски оквир за анализу методом коначних елемената. Предложени сурогат модели су креирани за један скуп параметара Хакслијевог модела мишића, па би будућа истраживања могла да укључе друге сурогат моделе за друге скупове параметара, који се односе на различите мутације мишићних протеина, чиме се може пратити утицај мутације на механички одзив леве коморе.

Литература

- [1] L. Ku, J. Feiger, M. Taylor, and L. Mestroni, “Familial Dilated Cardiomyopathy,” *Circulation*, vol. 108, no. 17. Ovid Technologies (Wolters Kluwer Health), Oct. 28, 2003. doi: 10.1161/01.cir.0000097493.70422.50.
- [2] B. Stojanovic, M. Svcevic, A. Kaplarevic-Malistic, R. J. Gilbert, and S. M. Mijailovich, “Multi-scale striated muscle contraction model linking sarcomere length-dependent cross-bridge kinetics to macroscopic deformation,” *Journal of Computational Science*, vol. 39. Elsevier BV, p. 101062, Jan. 2020. doi:10.1016/j.jocs.2019.101062.
- [3] B. S. Stojanovic, M. R. Svcevic, A. M. Kaplarevic-Malistic et al., “Coupling finite element and huxley models in multiscale muscle modeling,” 2015 IEEE 15th International Conference on Bioinformatics and Bioengineering (BIBE). IEEE, Nov. 2015. doi:10.1109/bibe.2015.7367674.
- [4] M. Ivanović, B. Stojanović, A. Kaplarević-Mališić, R. Gilbert, and S. Mijailovich, “Distributed multi-scale muscle simulation in a hybrid MPI–CUDA computational environment,” *Simulation*, vol. 92, no. 1. SAGE Publications, pp. 19–31, Dec. 11, 2015. doi:10.1177/0037549715620299.
- [5] M. Ivanović, A. Kaplarević-Mališić, B. Stojanović, M. Svičević, and S. M. Mijailovich, “Machine learned domain decomposition scheme applied to parallel multi-scale muscle simulation,” *The International Journal of High Performance Computing Applications*, vol. 33, no. 5. SAGE Publications, pp. 885–896, Mar. 12, 2019. doi:10.1177/1094342019833151.
- [6] S. Yan, X. Zou, M. Ilkhani, and A. Jones, “An efficient multiscale surrogate modelling framework for composite materials considering progressive damage based on artificial neural networks,” *Composites Part B: Engineering*, vol. 194. Elsevier BV, p. 108014, Aug. 2020. doi:10.1016/j.compositesb.2020.108014.
- [7] F. Ghavamian and A. Simone, “Accelerating multiscale finite element simulations of history-dependent materials using a recurrent neural network,” *Computer Methods in Applied Mechanics and Engineering*, vol. 357. Elsevier BV, p. 112594, Dec. 2019. doi:10.1016/j.cma.2019.112594.
- [8] F. Feyel, A multilevel finite element method (FE2) to describe the response of highly non-linear structures using generalized continua, *Comput. Methods Appl. Mech. Engrg.* 28–30 (2003) 3233–3244.
- [9] G. Dardis, “Heart anatomy cross-section,” *Kaleidoscope Fighting Lupus*, 05-Aug-2020. <https://kaleidoscopefightinglupus.org/lupus-body-precious-heart-cardiovascular-system/heart-anatomy-cross-section/> [Приступљено: 02-Dec-2022].

- [10] J. R. Mitchell and J.-J. Wang, “Expanding application of the Wiggers diagram to teach cardiovascular physiology,” *Advances in Physiology Education*, vol. 38, no. 2. American Physiological Society, pp. 170–175, Jun. 2014. doi: 10.1152/advan.00123.2013.
- [11] A. F. Huxley, “Muscle structure and theories of contraction.”, *Progress in biophysics and biophysical chemistry*, vol. 7, pp. 255–318, 1957, doi:10.1016/s0096-4174(18)30128-8.
- [12] A. M. Gordon, A. F. Huxley, and F. J. Julian, “The variation in isometric tension with sarcomere length in vertebrate muscle fibres”, *The Journal of Physiology*, vol. 184, no. 1, Art. no. 1, 1966, doi:10.1113/jphysiol.1966.sp007909.
- [13] P. J. Hunter, A. D. McCulloch, and H. E. D. J. ter Keurs, “Modelling the mechanical properties of cardiac muscle,” *Progress in Biophysics and Molecular Biology*, vol. 69, no. 2–3. Elsevier BV, pp. 289–331, Mar. 1998. doi:10.1016/s0079-6107(98)00013-3.
- [14] M. Lister, “The numerical solution of hyperbolic partial differential equations by the method of characteristics”, *Numerical Methods for Digital Computers*, pp. 165–179, 1960.
- [15] S. M. Mijailovich, J. J. Fredberg, and J. P. Butler, “On the theory of muscle contraction: filament extensibility and the development of isometric force and stiffness.”, *Biophys. J.*, vol. 71, no. 3, pp. 1475–84, Sep. 1996.
- [16] M. Kojic and K. J. Bathe, “*Inelastic Analysis of Solids and Structures*”, *Inelastic Analysis of Solids and Structures*, 2005.
- [17] K. J. Bathe, “*Finite element procedures*. Englewood Cliffs, NJ: Prentice-Hall; 1996.
- [18] B. Stojanovic, M. Kojic, M. Rosic, C. P. Tsui, and C. Y. Tang, “An extension of Hill’s three-component model to include different fibre types in finite element modelling of muscle,” *Int. J. Numer. Methods Eng.*, vol. 71, no. 7, pp. 801–817, 2007.
- [19] S. M. Mijailovich, B. Stojanovic, M. Kojic, A. Liang, V. J. Wedeen, and R. J. Gilbert, “Derivation of a finite-element model of lingual deformation during swallowing from the mechanics of mesoscale myofiber tracts obtained by MRI,” *Journal of Applied Physiology*, vol. 109, no. 5. American Physiological Society, pp. 1500–1514, Nov. 2010. doi:10.1152/jappphysiol.00493.2010.
- [20] B. Milićević, M. Ivanović, B. Stojanović, M. Milošević, M. Kojić, and N. Filipović, “Huxley muscle model surrogates for high-speed multi-scale simulations of cardiac contraction”, *Computers in Biology and Medicine*, vol. 149, p. 105963, Oct. 2022, doi:10.1016/j.compbimed.2022.105963.
- [21] G. A. Holzapfel and R. W. Ogden, “Constitutive modelling of passive myocardium: a structurally based framework for material characterization,” *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 367, no. 1902. The Royal Society, pp. 3445–3475, Sep. 13, 2009. doi:10.1098/rsta.2009.0091.

- [22] J. M. Guccione, A. D. McCulloch, and L. K. Waldman, “Passive Material Properties of Intact Ventricular Myocardium Determined From a Cylindrical Model,” *Journal of Biomechanical Engineering*, vol. 113, no. 1. ASME International, pp. 42–55, Feb. 01, 1991. doi:10.1115/1.2894084.
- [23] E. McEvoy, G. A. Holzapfel, and P. McGarry, “Compressibility and Anisotropy of the Ventricular Myocardium: Experimental Analysis and Microstructural Modeling,” *Journal of Biomechanical Engineering*, vol. 140, no. 8. ASME International, May 24, 2018. doi:10.1115/1.4039947.
- [24] M. Kojic, M. Milosevic, B. Milicevic, V. Geroski, V. Simic, D. Trifunovic, G. Stankovic and N. Filipovic, “Computational model for heart tissue with direct use of experimental constitutive relationships,” *Journal of the Serbian Society for Computational Mechanics*, vol. 15., no. 1, pp 1-23, 2021. doi:10.24874/jsscm.2021.15.01.01.
- [25] M. Kojic, M. Milosevic, A. Ziemys and N. Filipovic, “Computational Models in Biomedical Engineering - Finite Element Models Based on Smeared Physical Fields: Theory, Solutions, and Software,” Elsevier, 2022. doi:10.1016/C2020-0-03703-5
- [26] M. Kojic, M. Milosevic, V. Simic, B. Milicevic, V. Geroski, S. Nizzero, A. Ziemys and N. Filipovic, “Smeared Multiscale Finite Element Models for Mass Transport and Electrophysiology Coupled to Muscle Mechanics,” *Frontiers in Bioengineering and Biotechnology*, vol. 7. Frontiers Media SA, Dec. 10, 2019. doi:10.3389/fbioe.2019.00381.
- [27] M. Anić, S. Savić, A. Milovanović, M. Milošević, B. Milićević, V. Simić and N. Filipović, “Solution of Fluid Flow Through the Left Heart Ventricle,” *Applied Engineering Letters : Journal of Engineering and Applied Sciences*, vol. 5, no. 4. Faculty of Philology, University of Belgrade, pp. 120–125, 2020. doi:10.18485/aeletters.2020.5.4.2.
- [28] M. Böhl, R. Weikert, and C. Weichert, “A coupled electromechanical model for the excitation-dependent contraction of skeletal muscle”, *Journal of the Mechanical Behavior of Biomedical Materials*, vol. 4, no. 7, Art. no. 7, 2011, doi:10.1016/j.jmbbm.2011.04.017.
- [29] O. Röhrle, J. B. Davidson, and A. J. Pullan, “A physiologically based, multi-scale model of skeletal muscle structure and function”, *Frontiers in Physiology*, vol. 3 SEP, 2012, doi:10.3389/fphys.2012.00358.
- [30] M. Ben Belgacem, B. Chopard, J. Borgdorff, M. Mamoński, K. Rycerz, and D. Harezlak, “Distributed multiscale computations using the MAPPER framework”, presented at the *Procedia Computer Science*, 2013, vol. 18, pp. 1106–1115. doi:10.1016/j.procs.2013.05.276.

- [31] J. Borgdorff et al., “Distributed multiscale computing with MUSCLE 2, the Multiscale Coupling Library and Environment”, *Journal of Computational Science*, vol. 5, no. 5, Art. no. 5, 2014, doi:10.1016/j.jocs.2014.04.004.
- [32] L. Chin, P. Yue, J. J. Feng, and C. Y. Seow, “Mathematical simulation of muscle cross-bridge cycle and force-velocity relationship”, *Biophysical Journal*, vol. 91, no. 10, Art. no. 10, 2006, doi:10.1529/biophysj.106.092510.
- [33] M. Böl, “Micromechanical modelling of skeletal muscles: From the single fibre to the whole muscle”, *Archive of Applied Mechanics*, vol. 80, no. 5, Art. no. 5, 2010, doi:10.1007/s00419-009-0378-y.
- [34] J. Bestel, “Modèle différentiel de la contraction musculaire contrôlée”, *Application au système cardiovasculaire*, 2000.
- [35] T. Heidlauf and O. Röhrle, “Modeling the chemoelectromechanical behavior of skeletal muscle using the parallel open-source software library openCMISS”, *Computational and Mathematical Methods in Medicine*, vol. 2013, 2013, doi:10.1155/2013/517287.
- [36] M. Kojic, S. Mijailovic, and N. Zdravkovic, “Modelling of muscle behaviour by the finite element method using Hills three-element model”, *International Journal for Numerical Methods in Engineering*, vol. 43, no. 5, Art. no. 5, 1998, doi:10.1002/(SICI)1097-0207(19981115)43:5<941::AID-NME435>3.0.CO;2-3.
- [37] J. W. Fernandez, M. L. Buist, D. P. Nickerson, and P. J. Hunter, “Modelling the passive and nerve activated response of the rectus femoris muscle to a flexion loading: A finite element framework”, *Medical Engineering and Physics*, vol. 27, no. 10, Art. no. 10, 2005, doi:10.1016/j.medengphy.2005.03.009.
- [38] P. J. Basser, J. Mattiello, and D. LeBihan, “MR diffusion tensor spectroscopy and imaging”, *Biophysical Journal*, vol. 66, no. 1, Art. no. 1, 1994, doi:10.1016/S0006-3495(94)80775-1.
- [39] J. O. Dada and P. Mendes, “Multi-scale modelling and simulation in systems biology”, *Integrative Biology*, vol. 3, no. 2, Art. no. 2, 2011, doi:10.1039/c0ib00075b.
- [40] P. M. Slood and A. G. Hoekstra, “Multi-scale modelling in computational biomedicine”, *Briefings in Bioinformatics*, vol. 11, no. 1, Art. no. 1, 2009, doi:10.1093/bib/bbp038.
- [41] H. El Makssoud et al., “Multiscale modeling of skeletal muscle properties and experimental validations in isometric conditions”, *Biological Cybernetics*, vol. 105, no. 2, Art. no. 2, 2011, doi:10.1007/s00422-011-0445-7.
- [42] J. M. Guccione, A. D. McCulloch, and L. K. Waldman, “Passive material properties of intact ventricular myocardium determined from a cylindrical model”, *Journal of Biomechanical Engineering*, vol. 113, no. 1, Art. no. 1, 1991, doi:10.1115/1.2894084.

- [43] A. Torelli, “Study of a mathematical model for muscle contraction with deformable elements”, *Rend Sem Mat Univ Politec Torino*, vol. 55, no. 3, Art. no. 3, 1997.
- [44] A. Zhang, Z. C. Lipton, M. Li, and A. J. Smola, “Dive into Deep Learning”, p. 1022.
- [45] M. P. Deisenroth, A. A. Faisal, and C. S. Ong, “Mathematics for Machine Learning”, p.412.
- [46] Y. LeCun et al., “Backpropagation applied to handwritten zip code recognition”, *Neural Computation*, vol. 1, no. 4, Art. no. 4, 1989.
- [47] X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks”, *Journal of Machine Learning Research*, vol. 9, pp. 249–256, 2010.
- [48] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, “Understanding deep learning requires rethinking generalization.” *arXiv*, Feb. 26, 2017. <http://arxiv.org/abs/1611.03530>
- [49] S. Wager, S. Wang, and P. Liang, “Dropout training as adaptive regularization”, presented at the Advances in Neural Information Processing Systems, 2013.
- [50] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting”, *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014.
- [51] S. Wang and C. Manning, “Fast dropout training”, *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, no. 2, Art. no. 2, 2013.
- [52] N. Srivastava, “Improving neural networks with dropout”, *Improving Neural Networks with Dropout*, 2013.
- [53] S. J. Reddi, S. Kale, and S. Kumar, “On the convergence of Adam and beyond”, 6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings, 2018.
- [54] T. Tan, S. Yin, K. Liu, and M. Wan, “On the convergence speed of AMSGRAD and beyond”, presented at the Proceedings - International Conference on Tools with Artificial Intelligence, ICTAI, 2019, vol. 2019-November, pp. 464–470. doi:10.1109/ICTAI.2019.00071.
- [55] L. Liu et al., “On the Variance of the Adaptive Learning Rate and Beyond”, no. arXiv:1908.03265. *arXiv*, Oct. 25, 2021. <http://arxiv.org/abs/1908.03265>
- [56] G. Goh, “Why Momentum Really Works.,” *Distill*, vol. 2, no. 4. Distill Working Group, Apr. 04, 2017. doi:10.23915/distill.00006.
- [57] N. Qian, “On the momentum term in gradient descent learning algorithms,” *Neural Networks*, vol. 12, no. 1, pp. 145–151, Jan. 1999, doi:[10.1016/S0893-6080\(98\)00116-6](https://doi.org/10.1016/S0893-6080(98)00116-6).

- [58] B. T. Polyak, “Some methods of speeding up the convergence of iteration methods,” *USSR Computational Mathematics and Mathematical Physics*, vol. 4, no. 5, pp. 1–17, Jan. 1964, doi:[10.1016/0041-5553\(64\)90137-5](https://doi.org/10.1016/0041-5553(64)90137-5).
- [59] W. Su, S. Boyd, and E. J. Candes, “A Differential Equation for Modeling Nesterov’s Accelerated Gradient Method: Theory and Insights.” arXiv, Oct. 27, 2015. <http://arxiv.org/abs/1503.01243>
- [60] M. Raissi, P. Perdikaris, and G. E. Karniadakis, “Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations,” *Journal of Computational Physics*, vol. 378. Elsevier BV, pp. 686–707, Feb. 2019. doi:10.1016/j.jcp.2018.10.045.
- [61] L.C. Jain, L.R. Medsker, *Recurrent Neural Networks: Design and Applications*, first ed., CRC Press, Inc., Boca Raton, FL, USA, 1999.
- [62] R. Pascanu, T. Mikolov, and Y. Bengio, “On the difficulty of training Recurrent Neural Networks.” arXiv, 2012. doi:10.48550/ARXIV.1211.5063.
- [63] R. Dey and F. M. Salem, “Gate-variants of Gated Recurrent Unit (GRU) neural networks,” 2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS). IEEE, Aug. 2017. doi:10.1109/mwscas.2017.8053243.
- [64] Y. Yu, X. Si, C. Hu, and J. Zhang, “A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures,” *Neural Computation*, vol. 31, no. 7. MIT Press - Journals, pp. 1235–1270, Jul. 2019. doi:10.1162/neco_a_01199.
- [65] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory”, *Neural Computation*, vol. 9, no. 8, Art. no. 8, 1997, doi:10.1162/neco.1997.9.8.1735.
- [66] J. R. A. Moniz and D. Krueger, “Nested LSTMs”, no. arXiv:1801.10308. arXiv, Jan. 31, 2018. <http://arxiv.org/abs/1801.10308>
- [67] *Understanding Convolutions - colah's blog*.
<https://colah.github.io/posts/2014-07-Understanding-Convolutions/>
[Приступљено: 01.12.2022.]
- [68] S. Bai, J. Z. Kolter, and V. Koltun, “An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling”, no. arXiv:1803.01271. arXiv, Apr. 19, 2018. <http://arxiv.org/abs/1803.01271>
- [69] A. van den Oord et al., “WaveNet: A Generative Model for Raw Audio”, no. arXiv:1609.03499. arXiv, Sep. 19, 2016. <http://arxiv.org/abs/1609.03499>
- [70] B. Milicevic et al., “Integration of Surrogate Huxley Muscle Model into Finite Element Solver for Simulation of the Cardiac Cycle”, in 2022 44th Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC),

Glasgow, Scotland, United Kingdom, Jul. 2022, pp. 3943–3946.
doi:10.1109/EMBC48229.2022.9870995.

- [71] D. Panozzo and A. Jacobson, “LIBIGL: A C++ library for geometry processing without a mesh data structure,” 2014.

Додатак А

За имплементацију неуронских мрежа коришћена је библиотека *Keras*, са *TensorFlow*-ом као *backend*-ом. *TensorFlow* је библиотека отвореног кода, која служи за нумеричке прорачуне. Има флексибилну архитектуру која омогућава развој софтвера на различитим платформама, од десктоп рачунара до кластера и сервера, као и мобилним уређајима. Могуће је извршавање кода на централним, графичким и тензор процесним јединицама. *TensorFlow* је развијен од стране тима *Google Brain* у оквиру *Google AI* организације. Библиотека пружа јаку подршку за машинско учење, дубинско учење и због флексибилног језгра библиотека се користи у многим другим научним доменима. Међу подржаним језицима су *Python*, *R*, *C*, *C++*, *Java*, *Go*. *Keras* је библиотека отвореног кода, намењена брзом експериментисању са дубоким неуронским мрежама, иницијално развијана у оквиру пројекта *ONEIROS (Open-ended Neuro-Electronic Intelligent Robot Operating System)*. *Keras* значи рог на грчком. У Хомеровој Одисеји, духови снова (*Oneiroi*, једина *Oneiros*) се деле на оне, који долазе на Земљу кроз капију од слоноваче, и обмањују лажним визијама и на оне који долазе кроз капију од рогова и који најављују будућност која ће доћи. У питању је игра речима *κέρας* (рог) / *κράϊνω* (испунити), и *ἐλέφας* (слоновача) / *ἐλεφαίρομαι* (обманути). Библиотека *Keras* је направљена да буде интерфејс, а не самостални фрејмворк за машинско учење, нуди висок ниво апстракције, са којим је лакше развити моделе учења без обзира на то која је библиотека коришћена као подлога, односно *backend*. *Keras* може да ради са библиотекама као што су *TensorFlow*, *Microsoft Cognitive Toolkit* и *Theano*. Садржи разне имплементације често коришћених градивних блокова као што су: слојеви, објективи, функције активације, оптимизациони алгоритми и слично. *Keras* има подршку за различите врсте неуронских мрежа, на пример, рекурентне и конволуционе мреже, као и за комбиновање различитих врста мрежа. *Keras* може да ради на централним и графичким процесним јединицама. Ова библиотека може бити коришћена у *Python*-у и *R*-у.

У наставку ће бити дати листинзи написаних програмских кодова за креирање и обучавање GRU мреже, као и за изометријску контракцију са неуронским мрежама подржаним физичким законима. Наведени и остали кодови за креирање и обучавање неуронских мрежа, заједно са кодовима за интеграцију сурогат модела у фрејмворк за анализу методом коначних елемената се могу пронаћи у следећим *github* репозиторијумима:

- <https://github.com/BogdanM1/surro-muscle>
- https://github.com/BogdanM1/pinn_huxley

У листингу за креирање и обучавање GRU мреже се налази код којим се најпре позива скрипта за препроцесирање (*initialize.py*), затим се подаци конвертују у временске серије, конструише се неуронска мрежа и позива процедура за обуку.

Креирање и обучавање GRU мреже

```

from tensorflow.keras.callbacks import ModelCheckpoint, EarlyStopping
from keras.layers import Dense, Input, Dropout, Flatten, GRU
from keras.models import Model, Sequential
from numpy.random import seed

commands = open("initialize.py").read()
exec(commands)
model_path = '../models/model.h5'
X = []
Y = []
for i in itertools.chain(np.setdiff1d(range(ntrains,ntraine),range(ntrains+3,ntraine,4))):
    indices = data['testid'].isin([i])
    for x in InputToTimeSeries(data_scaled[indices][:,time_series_feature_columns],
np.array(data.loc[indices,'converged'])):
        X.append(x)
    for y in InputToTimeSeries(data_scaled[indices][:,target_columns],
np.array(data.loc[indices,'converged'])):
        Y.append(y[-1])
    data = data[indices != True]
    data_scaled = data_scaled[indices != True]
X_val = []
Y_val = []
for i in itertools.chain(range(ntrains+3,ntraine,4)):
    indices = data['testid'].isin([i])
    for x in InputToTimeSeries(data_scaled[indices][:, time_series_feature_columns],
np.array(data.loc[indices,'converged'])):
        X_val.append(x)
    for y in InputToTimeSeries(data_scaled[indices][:, target_columns],
np.array(data.loc[indices,'converged'])):
        Y_val.append(y[-1])
    data = data[indices != True]
    data_scaled = data_scaled[indices != True]
values_prev = [sample[-1][-2:] for sample in X]
values_prev_val = [sample[-1][-2:] for sample in X_val]
X = np.array(X)
Y = np.array(Y)
Y -= np.array(values_prev)
Y *= stress_scale
X_val = np.array(X_val)
Y_val = np.array(Y_val)
Y_val -= np.array(values_prev_val)
Y_val *= stress_scale
lecun_normal = keras.initializers.lecun_normal(seed= seed)
orthogonal = keras.initializers.Orthogonal(seed=_seed)
glorot_uniform = keras.initializers.glorot_uniform(seed=_seed)
i = Input(shape=(time_series_steps, len(time_series_feature_columns)), name='input_layer')
o = GRU(128, return_sequences=True, activation='selu', kernel_initializer=lecun_normal,
recurrent_initializer=orthogonal, name='gru1')(i)
o = GRU(256, return_sequences=True, activation='selu', kernel_initializer=lecun_normal,
recurrent_initializer=orthogonal, name='gru2')(o)
o = GRU(256, return_sequences=True, activation='selu', kernel_initializer=lecun_normal,
recurrent_initializer=orthogonal, name='gru3')(o)
o = GRU(128, return_sequences=True, activation='selu', kernel_initializer=lecun_normal,
recurrent_initializer=orthogonal, name='gru4')(o)
o = GRU(128, return_sequences=True, activation='selu', kernel_initializer=lecun_normal,
recurrent_initializer=orthogonal, name='gru5')(o)
o = Flatten()(o)
o = Dense(2, name='output_layer')(o)

model = Model(inputs = [i], outputs=[o])
model.compile(loss=loss, optimizer=optimizer)
print(model.summary())
modelcheckpoint = ModelCheckpoint(model_path, monitor = 'val_loss', save_best_only = True)
earlystopping = EarlyStopping(monitor='val_loss', restore_best_weights=True, patience=500)
model.fit(X, Y, epochs = 50000, batch_size = 16384, validation_data = (X_val, Y_val),
verbose=2, callbacks = [tensorboard, modelcheckpoint, earlystopping])

```

У листингу за креирање и обучавање неуронске мреже подржане Хакслијевом једначином за изометријску контракцију су наведени фиксни параметри Хакслијевог материјалног модела, функција за корекцију Гордоновом кривом, функције за стопу

закачињања и откачињања миозинских глава, затим је креиран функционал који представља сурогат модел, дефинисане су функције губитка (резидуал диференцијалне једначине L1 и резидуал почетног услова I1), генерисани су улазни подаци и позвана је процедура за обуку.

Креирање и обучавање неуронске мреже подржане Хакслијевом једначином – Изометријски случај

```
import numpy as np
import pandas as pd
import sciann as sn
from sciann.utils.math import diff, sign
''' fixed parameters '''
TOL = 1e-4
f1_0 = 43.3
h = 15.6
g1 = 10.0
g2 = 209.0
fzah = 4.0
L0 = 1100.0
dt = 1e-3
grdstretch = [0.6, 0.8, 0.95, 1.0, 1.64, 5.0]
grdstress = [0.0, 0.782, 1.0, 1.0, 0.0, 0.0]
''' fixed parameters '''

def lininterp(x, x0, x1, y0, y1):
    return (y0 + (x-x0)*(y1 - y0)/(x1 - x0))

def gordon_correction(stretch, n):
    cor = (stretch >= grdstretch[0] and stretch < grdstretch[1])*lininterp(stretch,
grdstretch[0],grdstretch[1], grdstress[0],grdstress[1])
    cor+=( stretch >= grdstretch[1] and stretch < grdstretch[2])*lininterp(stretch,
grdstretch[1],grdstretch[2], grdstress[1],grdstress[2])
    cor+=( stretch >= grdstretch[2] and stretch < grdstretch[3])*lininterp(stretch,
grdstretch[2],grdstretch[3], grdstress[2],grdstress[3])
    cor+=( stretch >= grdstretch[3] and stretch < grdstretch[4])*lininterp(stretch,
grdstretch[3],grdstretch[4], grdstress[3],grdstress[4])
    return (cor > n)*(cor - n)

def f(x,a):
    return (x >= .0 and x <= h)*(f1_0*a*x/h)

def g(x):
    return ((x < .0)*g2 +
            (x >= .0 and x <= h)*(g1*x/h) +
            (x > h)*(fzah*g1*x/h))

x = sn.Variable('x')
t = sn.Variable('t')
n = sn.Functional('n', [x,t], 8*[20], 'tanh')
L1 = (diff(n, t) - (1.0-n)*f(x,1.0) + n*g(x) ) * (1+sign(n)) * 0.5
I1 = (t < TOL)*n
I2 = (1-sign(n))*n
model = sn.SciModel([x,t], [L1, I1, I2])
x_train = np.arange(-20.8,63,0.65)
t_train = np.linspace(0, .5, 500)
x_train, t_train = np.meshgrid( x_train, t_train )
h = model.train([x_train, t_train], ['zeros','zeros', 'zeros'], learning_rate=1e-4,
batch_size=512, epochs=30000, stop_loss_value=1e-9, adaptive_weights={'method':'NTK',
'freq':300}, verbose=2, save_weights = {'path':'../models/isom-best_model',
'best':True,'freq':1})
```

У табелама А.1 и А.2 су дата поређења вредности напона у нумеричким експериментима добијених сурогат моделом заснованим на GRU неуронској мрежи и оригиналним Хакслијевим моделом. Сви резултати су добијени коришћењем *Mexie* солвера за анализу методом коначних елемената. Приказани су редни бројеви експеримената и степени корелације између вредности напона добијених оригиналним

и сурогат моделом. Укупан број експеримената је 165, од тога већина нумеричких експеримената је коришћена за обуку (120), сваки четврти експеримент је коришћен за валидацију (39), а последњих 6 експеримената је коришћено за тестирање неуронске мреже.

Табела А.1 Поређење вредности напона у нумеричким експериментима (из скупа за обуку) добијених сурогат моделом заснованим на GRU неуронској мрежи и оригиналним Хакслијевим моделом

No.	CORR (напон)	No.	CORR (напон)	No.	CORR (напон)	No.	CORR (напон)	No.	CORR (напон)
1	0.999998	33	0.999964	65	0.999993	97	0.999991	129	0.999914
2	0.999998	34	0.999976	66	0.999537	98	1.000000	130	0.999994
3	0.999961	35	0.999969	67	0.996930	99	0.999999	131	0.999975
5	0.999893	37	0.999935	69	0.999170	101	0.999998	133	0.999900
6	0.999961	38	0.999909	70	0.998971	102	0.999994	134	0.999955
7	0.999990	39	0.999877	71	0.999999	103	0.999987	135	0.999681
9	0.999988	41	0.999862	73	0.999987	105	0.999994	137	0.999978
10	0.999904	42	0.999840	74	0.999998	106	0.999720	138	0.999939
11	0.999710	43	0.999762	75	0.999999	107	0.997694	139	0.999929
12	0.998714	45	0.999484	77	0.999999	109	0.999811	141	0.999991
13	0.999994	46	0.999997	78	0.999994	110	0.999929	142	0.999993
14	0.999995	47	0.999997	79	0.999997	111	0.999981	143	0.999982
15	0.999950	49	0.999997	81	0.999999	113	0.999744	145	0.999814
17	0.999875	50	0.999997	82	0.999999	114	0.999081	146	0.999988
18	0.999881	51	0.999995	83	0.999998	115	0.999982	147	0.999994
19	0.999582	53	0.999995	85	0.999943	117	0.999975	149	0.999998
21	0.989298	54	0.999995	86	0.999992	118	0.999973	150	0.999993
22	0.999356	55	0.999994	87	0.999876	119	0.999936	151	0.999995
23	0.999793	57	0.999995	89	0.999992	121	0.999865	153	0.999973
25	0.999554	58	0.999995	90	0.999992	122	0.999892	154	0.999993
26	0.999765	59	0.999994	91	1.000000	123	0.999938	155	0.999993
27	0.999936	61	0.999996	93	0.999998	125	0.999936	157	0.999992
29	0.999941	62	0.999995	94	0.999989	126	0.999986	158	0.999904
30	0.999949	63	0.999996	95	0.999994	127	0.999933	159	0.999990
31	0.999956								

Нумерички експерименти са изотоничном контракцијом су обележени бројевима од 1 до 45, експерименти са брзим ослобађањем су обележени бројевима од 46 до 65, експерименти са задатим силама су обележени бројевима од 66 до 105, а експерименти са задатим померањима су обележени бројевима од 106 до 165. Из приложених табела се види да је сурогат модел дао веома слична решења као оригинални модел у свим нумеричким експериментима. Већа сличност је постигнута у експериментима који су коришћени за обуку, него у експериментима коришћеним за тестирање, што је очекивано, али без обзира на смањење прецизности предвиђених напона у случајевима за тестирање, односно без обзира на постојање генерализационог јаза, предвиђене вредности су и у овим случајевима довољно сличне оригиналним вредностима.

Табела А.2 Поређење вредности напона у нумеричким експериментима (из скупа за валидацију и обуку) добијених сурогат моделом заснованим на *GRU* неуронској мрежи и оригиналним Хакслијевим моделом

No.	CORR (напон)	No.	CORR (напон)	No.	CORR (напон)	No.	CORR (напон)
Скуп за валидацију						Скуп за тестирање	
4	0.999898	60	0.999996	112	0.999684	160	0.999743
8	0.999932	64	0.999995	116	0.998306	161	0.999983
16	0.999702	68	0.997657	120	0.999515	162	0.999962
20	0.999943	72	0.999998	124	0.999797	163	0.999813
24	0.999311	76	0.999986	128	0.999923	164	0.999995
28	0.999919	80	0.999993	132	0.999843	165	0.936953
32	0.999953	84	0.999998	136	0.999901		
36	0.999955	88	0.999995	140	0.999955		
40	0.999846	92	0.999988	144	0.998343		
44	0.999603	96	0.999999	148	0.999941		
48	0.999997	100	0.999993	152	0.997014		
52	0.999995	104	0.999987	156	0.999940		
56	0.999994	108	0.999960				

Биографија кандидата

Богдан Милићевић је рођен 20.08.1992. године у Крагујевцу. Основну школу "Драгиша Луковић-Шпанац", у Крагујевцу, завршио је 2007. године. Средњу школу Прву крагујевачку гимназију, завршио је 2011. године.

Основне академске студије на Природно-математичком факултету у Крагујевцу уписао је 2011. године. Завршни рад на тему "Вештачка интелигенција за препознавање текста на сликама" одбранио је 2015. године чиме је стекао звање дипломирани информатичар. Мастер студије на Природно-математичком факултету у Крагујевцу уписао је 2015. године. Мастер рад на тему "Имплементација вишескалног модела дифузије на паралелним архитектурама" одбранио је 2016. године, чиме је стекао звање мастер информатичар.

Докторске академске студије уписао је школске 2017/2018 године, на Факултету инжењерских наука у Крагујевцу, под менторством др Ненада Филиповића, ред. проф. Факултета инжењерских наука Универзитета у Крагујевцу.

Запослен је на Факултету инжењерских наука Универзитета у Крагујевцу као истраживач-сарадник. Бави се рачунарским симулацијама, моделирањем и машинским учењем. У оквиру истраживања учествовао је на неколико пројеката Horizont2020 (SILICOFCM (бр. 777204), SGABU (бр.952603)). Учествује у реализацији наставе на Факултету инжењерских наука Универзитета у Крагујевцу на предметима: Практикум из основа рачунарске технике, Основи рачунарске технике 2, Рачунарске основе интернета, Паралелни рачунарски системи, Архитектура рачунарских система, Микропроцесорски системи и Рачунарски алати.

ИЗЈАВА АУТОРА О ОРИГИНАЛНОСТИ ДОКТОРСКЕ ДИСЕРТАЦИЈЕ

Изјављујем да докторска дисертација под насловом:

Суругат модели мишића засновани на вештачким неуронским мрежама

са применом у анализи методом коначних елемената

представља *оригинално ауторско дело* настало као резултат *сопственог истраживачког рада*.

Овом Изјавом такође потврђујем:

- да сам *једини аутор* наведене докторске дисертације,
- да у наведеној докторској дисертацији *нисам извршио/ла повреду* ауторског нити другог права интелектуалне својине других лица,

У Крагујевцу, 5.12.2022. године,

Munirbat Bojars

потпис аутора

Образац 2

**ИЗЈАВА АУТОРА О ИСТОВЕТНОСТИ ШТАМПАНЕ И ЕЛЕКТРОНСКЕ ВЕРЗИЈЕ
ДОКТОРСКЕ ДИСЕРТАЦИЈЕ**

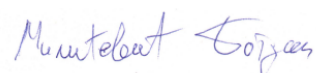
Изјављујем да су штампана и електронска верзија докторске дисертације под насловом:

Сурогат модели мишића засновани на вештачким неуронским мрежама

са применом у анализи методом коначних елемената

истоветне.

У Крагујевцу, 5.12.2022. године,



потпис аутора

ИЗЈАВА АУТОРА О ИСКОРИШЋАВАЊУ ДОКТОРСКЕ ДИСЕРТАЦИЈЕ

Ја, Богдан Милићевић

дозвољавам

не дозвољавам

Универзитетској библиотеци у Крагујевцу да начини два трајна умножена примерка у електронској форми докторске дисертације под насловом:

Суругат модели мишића засновани на вештачким неуронским мрежама

са применом у анализи методом коначних елемената

и то у целини, као и да по један примерак тако умножене докторске дисертације учини трајно доступним јавности путем дигиталног репозиторијума Универзитета у Крагујевцу и централног репозиторијума надлежног министарства, тако да припадници јавности могу начинити трајне умножене примерке у електронској форми наведене докторске дисертације путем *преузимања*.

Овом Изјавом такође

дозвољавам

не дозвољавам¹

¹ Уколико аутор изабере да не дозволи припадницима јавности да тако доступну докторску дисертацију користе под условима утврђеним једном од *Creative Commons* лиценци, то не искључује право припадника јавности да наведену докторску дисертацију користе у складу са одредбама Закона о ауторском и сродним правима.

припадницима јавности да тако доступну докторску дисертацију користе под условима утврђеним једном од следећих *Creative Commons* лиценци:

- 1) Ауторство
- 2) Ауторство - делити под истим условима
- 3) Ауторство - без прерада
- 4) Ауторство - некомерцијално
- 5) Ауторство - некомерцијално - делити под истим условима
- 6) Ауторство - некомерцијално - без прерада²

У Крагујевцу _____, 5.12.2022. године,



потпис аутора

² Молимо ауторе који су изабрали да дозволе припадницима јавности да тако доступну докторску дисертацију користе под условима утврђеним једном од *Creative Commons* лиценци да заокруже једну од понуђених лиценци. Детаљан садржај наведених лиценци доступан је на: <http://creativecommons.org.rs/>