

Примена кubernetesа и машинског учења у развоју система за временску прогнозу

Application of Kubernetes and Machine Learning in the Development of a Weather Forecasting System

Марко Василић, Факултет техничких наука, Нови Сад

Област – ЕЛЕКТРОТЕХНИКА И РАЧУНАРСТВО

Кратак садржај – Овај рад представља развој система за предвиђање температуре у реалном времену који комбинује технологије Кubernetesа, Докера и Long Short-Term Memory (ЛСТМ) неуронских мрежа. Архитектура система заснована је на микросервисима који обухватају сервис за предвиђање, сервис за управљање подацима, графички интерфејс и сервис за заказивање задатака. Систем је имплементиран и тестиран у Minikube Кubernetes кластеру, користећи PostgreSQL базу података за чување података и Streamlit библиотеку за визуелизацију резултата. LSTM модел је обучен на реалним метеоролошким подацима и омогућава прогнозу температуре за наредних 24 сата и 7 дана. Резултати показују да интеграција Кubernetesа и машинског учења омогућава ефикасну оркестрацију и скалабилност при дизајнирању система. Рад представља основу за даљи развој у правцу примене у клауд окружењима и интеграције са системима за аутоматско прикупљање и обраду података.

Кључне ријечи: Кubernetes, ЛСТМ, временска прогноза, машинско учење, Докер, Minikube, микросервисна архитектура.

Abstract – This paper presents the development of a scalable and automated system for real-time temperature prediction that integrates Kubernetes, Docker, and LSTM neural networks. The system architecture is based on a microservices model, including services for prediction, data management, visualization, and task scheduling. The implementation was carried out using Minikube Kubernetes clusters, with a PostgreSQL database for data storage and the Streamlit library for results visualization. The LSTM model was trained on meteorological data, enabling temperature forecasts for the next 24 hours and 7 days. The results demonstrate that the combination of Kubernetes and machine learning provides efficient orchestration and scalability in system design. This work establishes a foundation for further development toward cloud-based deployment and integration with automated data collection and processing systems.

Keywords: Kubernetes, LSTM, weather forecasting, machine learning, Docker, Minikube, microservices architecture.

НАПОМЕНА: Овај рад проистекао је из мастер рада чији је ментор била др Татјана Лончар-Турукало, ред. професор.

1. УВОД

Савремени технолошки развој донио је значајне иновације у начину на који се приступа обради великих количина података и изградњи скалабилних система. Једна од кључних области у том контексту је оркестрација контејнера, која омогућава ефикасно управљање комплексним апликацијама, те примјена машинског учења, које омогућава анализу и предвиђање појава заснованих на подацима.

Комбинација ових технологија представља основу за развој интелигентних и аутоматизованих система. У овом раду описан је процес изградње система за временску прогнозу који користи LSTM (енгл. Long Short-Term Memory) неуронске мреже за предвиђање температуре и Кubernetes као платформу за оркестрацију микросервисне архитектуре.

Главни циљ рада је да се прикаже на који начин Кubernetes омогућава лако управљање, скалирање и одржавање апликација заснованих на машинском учењу. У систему је примјењен Minikube кластер, који омогућава локално тестирање инфраструктуре, а модели су обучени на метеоролошким подацима. Развој оваквих система омогућава лакшу интеграцију аналитичких модела у продукциона окружења, што представља значајан корак ка примјени машинског учења у реалним системима.

У раду је описана коришћена методологија докер, и кubernetes платформе као и ЛСТМ мреже, након чега је описана имплементација система.

2. МЕТОДОЛОГИЈА

2.1. Докер

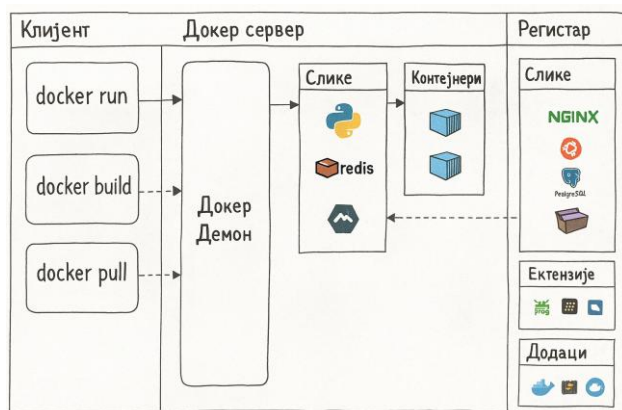
Докер је отворена платформа за развој, испоруку и извршавање апликација у контејнерима[1]. Он омогућава изолацију апликација од инфраструктуре, чиме се значајно убрзава развој и тестирање софтвера.

Докер користи архитектуру клијент–сервер, при чему Докер клијент комуницира са Докер демоном који управља контејнерима, сликама и мрежама преко REST API-ја.

Основни елементи Докера су:

- докер слике – шаблони из којих се покрећу контејнери;
- докер контејнери – извршне јединице које садрже апликацију и њене зависности;
- докер демон – сервис који извршава команде и управља контејнерима;
- докер регистар – складиште за слике, јавно или приватно.

Докер контејнери дијеле оперативни систем домаћина, што их чини лакшим и ефикаснијим у поређењу са виртуелним машинама. Захваљујући томе, апликације се могу лако преносити између различитих окружења – од локалне машине до клауда. Приказ Докер архитектуре дат на слици 1.



Слика 1. Илустрација Докер архитектуре

Употребом Докера постиже се: (1) портабилност - исти контејнер ради на свакој платформи; (2) изолација - свака апликација ради у свом окружењу, без конфликта са другим; (3) ефикасност ресурса - дели оперативни систем домаћина, па троши мање меморије и процесорског времена; (4) брза имплементација и скалирање: лако се креирају нови контејнери и систем се може проширити без застоја у раду.

Ове особине чине Докер кључном технологијом у савременим клауд архитектурама, као и у системима који користе Кубернетес за оркестрацију.

2.2. Кубернетес

Кубернетес је отворена платформа за оркестрацију контејнера, која омогућава аутоматизовано распоређивање, управљање и скалирање апликација [2]. Развијен од стране Google-а, данас представља индустријски стандард за управљање микросервисним системима у продукцији.

Главна идеја Кубернетеса је да се апликације покрећу у више независних контејнера који се групишу у *Pod*-ове, а затим управљају централизовано у кластеру. Сваки кластер се састоји од контролне равни (која

доноси одлуке и радних чворова који извршавају апликације.

Кубернетес обезбеђује: аутоматско скалирање апликација према оптерећењу, самоопоравак (рестарт падајућих контејнера), распоређивање без застоја (енгл. *rolling updates*), балансирање оптерећења и безбједно управљање конфигурацијама и тајнама.

Захваљујући овим особинама, Кубернетес омогућава да се системи машинског учења као што је овај лако одржавају, проширују и интегришу са другим сервисима.

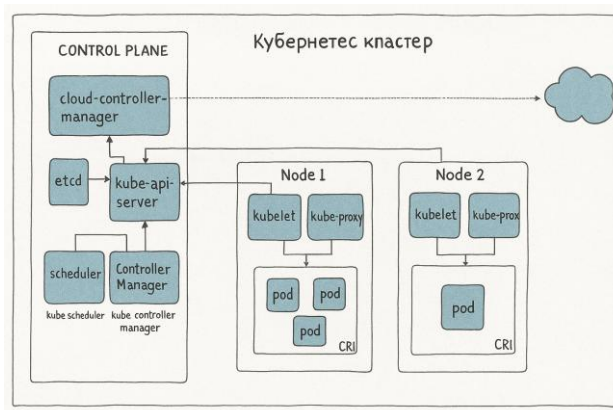
Архитектура Кубернетеса састоји се од неколико главних компоненти:

- *kube-apiserver* – прима и обрађује све захтјеве у кластеру;
- *etcd* – база података која чува стање кластера (key-value база);
- *kube-scheduler* – одређује расподјелу покретања *Pod*-ова по чворовима;
- *kube-controller-manager* – управља радом система и одржава га у жељеном стању;
- *kubelet* – агент који прати стање контејнера на сваком чвору.

Оваква архитектура омогућава декларативно управљање системом, гдје се у YAML конфигурационим фајловима описује жељено стање, а Кубернетес аутоматски обезбеђује да систем то стање задржи. Архитектура приказана на Сlici 2.

Примјена Кубернетеса у овом систему омогућава: лако распоређивање више микросервиса, аутоматско заказивање задатака, перзистентно чување података, једноставну интеграцију са PostgreSQL базом и визуелизацијом резултата.

Ове карактеристике чине Кубернетес идеалним окружењем за хостовање интелигентних система који комбинују машинско учење и обраду података у реалном времену.



Слика 2. Архитектура Кубернетес кластера

2.3. LSTM (Long Short-Term Memory)

Рекурентне неуронске мреже (РНН) [6], представљају врсту дубоких неуронских мрежа које су посебно

погодне за анализу секвенцијалних података, као што су временске серије, текст или звук. Кључна карактеристика РНН-а је повратна веза – излаз следећег или неког будућег слоја је улаз у неки ранији слој, на основу чега мрежа може „памтити“ информације из прошлости и користити их за будуће прогнозе.

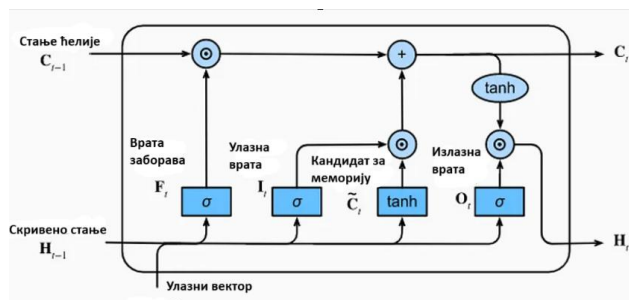
Ипак, класичне РНН мреже имају ограничења у раду са дугим секвенцама, јер током учења долази до појаве великог смањења или повећања градијената што утиче на поступак обуке и перформансе при дугорочном предвиђању градијената, што смањује њихову тачност при дугорочном предвиђању [3].

ЛСТМ (енгл. *Long Short-Term Memory*) [7] је специјализована варијанта RNN-а која уводи механизам меморијских ћелија и врата за контролу протока информација. Свака LSTM јединица садржи: врата заборавља – где се одлучује које старе информације ће бити обрисане; улазна врата – где се одређује које нове информације ће бити додате; излазна врата – која контролишу шта ће бити прослијеђено као излаз мреже.

Овом структуром ЛСТМ омогућава моделу да истовремено чува и краткорочне и дугорочне обрасце у подацима, што га чини изузетно погодним за временску прогнозу, гдје су историјски подаци релевантни за предвиђање будућих вредности. Приказ ЛСТМ архитектуре на слици 3.

За развој система коришћени су метеоролошки подаци за град Јена (Немачка) за 2024. годину, преузети са јавне базе података [4]. Након филтрирања, задржана су следећа обележја: температура (T), притисак (p), релативна влажност (rh), брзина вјетра (wv), густина ваздуха (ρ) и Vрст. Коришћено је 26207 записа (узорака) из базе. Модел је обучаван на нормализованим подацима. За оптимизацију је примјењен Adam оптимизатор, а функција губитка је средњеквадратна грешка (енгл. *Mean Squared Error*, MSE), која представља стандардну метрику за процјену перформанси регресионих модела.

Систем је имплементиран као микросервис у програмском језику Python, уз коришћење PyTorch библиотеке за развој и обуку LSTM неуронске мреже. Микросервис је упакован у Докер контејнер, а оркестрација је реализована помоћу Миникубе алата. За припрему и обраду података коришћене су *pandas* и *numpy* библиотеке.



Слика 3. ЛСТМ архитектура

3. ИМПЛЕМЕНТАЦИЈА СИСТЕМА

3.1. Архитектура решења

Циљ имплементације је развој система за временску прогнозу заснованог на микросервисној архитектури и покренутог у Кубернетес Minikube кластеру. Систем је подијељен у четири независна микросервиса који међусобно комуницирају путем REST API-ја:

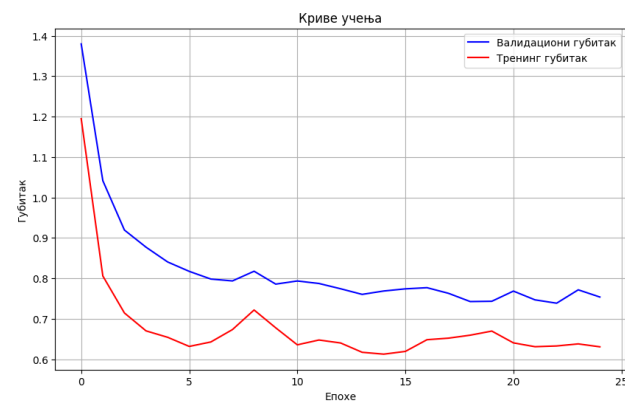
1. Сервис за предикцију – користи обучени LSTM модел за предвиђање температуре;
2. Сервис за повезивање – управља базом података, учитава CSV фајлове и иницира обуку модела;
3. Сервис за графички интерфејс – визуализује резултате предвиђања преко *Streamlit* библиотеке;
4. Сервис за заказивање задатака – аутоматски покреће процес тренинга модела сваки сат времена путем *CronJob* механизма у Кубернетесу.

Сви сервиси су запаковани у Докер контејнере, дефинисани помоћу *YAML* конфигурационих фајлова, и распоређени унутар *Minikube* кластера. Комуникација између њих реализована је преко унутрашњих сервисних ресурса у Кубернетесу.

3.3. ЛСТМ модел и предвиђање

Улазни подаци су нормализовани и структурирани у секвенце од по три дана, што омогућава LSTM моделу да „учи“ временске обрасце и предвиђа температуру за наредни период. LSTM модел се састоји од два слоја (64 и 32 јединице), при чему је у сврху регуларизације коришћен *dropout* (стопа постављена на 30%). Модел је обучаван уз Адам оптимизатор са брзином учења од 0.001. Модел врши двије врсте предвиђања: краткорочно – за наредних 24 сата (сатни интервал), дугорочно – за наредних 7 дана (дневни просјек).

Резултати тренинга показују конвергенцију грешке током 25 епоха, што указује на добру генерализацију модела. Коначни модел постиже средњу апсолутну грешку (MAE) од 0.971 на тест скупу, што представља задовољавајућу тачност предвиђања температуре. Критеријум заустављања био је фиксиран број епоха (25), без примјене раног заустављања на основу валидационе грешке. Приказ криве учења дат је на слици 4.



Слика 4. Криве учења модела за предикцију температуре.

3.4. Визуализација и интерфејс

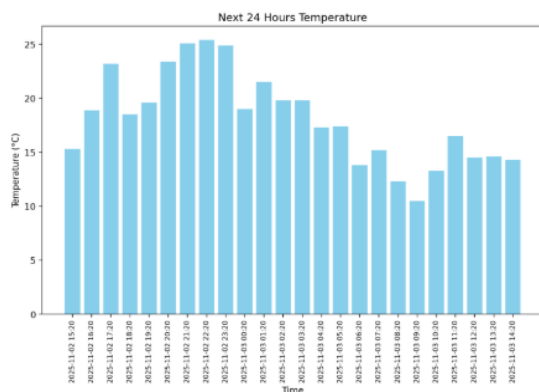
Кориснички интерфејс (UI) је развијен у *Streamlit*-у и омогућава приказ тренутне прогнозе у виду табеле и графикана. Подаци се аутоматски освјежавају при сваком позиву сервиса, што омогућава реално-временски преглед предвиђених температура (приказ на Слици 5).

Weather Prediction Dashboard

Next 24 Hours Forecast

Forecasted Temperatures for Next 24 Hours:

	Time	Temperature
0	2025-11-02 15:20	15.3
1	2025-11-02 16:20	18.9
2	2025-11-02 17:20	23.2
3	2025-11-02 18:20	18.5
4	2025-11-02 19:20	19.6
5	2025-11-02 20:20	23.4
6	2025-11-02 21:20	25.1
7	2025-11-02 22:20	25.4
8	2025-11-02 23:20	24.9
9	2025-11-03 00:20	19



Слика 5. Приказ графичког интерфејса

3.5. Интеграција и Кубернетес конфигурација

За сваки микросервис дефинисан је посебан Deployment и Service YAML фајл. PostgreSQL база је имплементирана као *StatefulSet*, за трајно складиштење података.

Сервис *scheduler* користи CronJob ресурс који сваких сат времена покреће процес поновне обуке модела, осигуравајући да предвиђања увијек користе најновије податке. Овај приступ обезбјеђује аутоматизацију, скалабилност и отпорност на грешке, што представља суштинску предност употребе Кубернетеса у оваквим системима.

4. ЗАКЉУЧАК

У овом раду приказан је развој система за временску прогнозу заснованог на ЛСТМ моделу и Кубернетес оркестрацији. Комбинацијом Докера, Кубернетеса и машинског учења постигнута је аутоматизација, скалабилност и поузданост система.

Систем је лако проширив и погодан за даљи развој у клауд окружењима, што представља основу за будуће интеграције интелигентних апликација у реалним сценаријима.

5. ЛИТЕРАТУРА

- [1] Docker (n.d.). Docker Documentation. Преузето 27.10.2025. са: <https://docs.docker.com/get-started/>
- [2] Kubernetes (n.d.). Kubernetes Documentation. Преузето 27.10.2025. са: <https://kubernetes.io/docs/home>
- [3] Ralf C. Staudemeyer, Eric Rothstein Morris (2019). Understanding LSTM -- a tutorial into Long Short-Term Memory Recurrent Neural Networks. Преузето 27.10.2025. са: <https://arxiv.org/abs/1909.09586>
- [4] Max Planck Institute for Biogeochemistry (n.d.). Weather dataset. Преузето 27.10.2025. са: <https://www.bgc-jena.mpg.de/wetter/>
- [5] Github репозиторијум, са кодом. Преузето 27.10.2025. са: <https://github.com/MarkoVasilic/WeatherForecastingMinikube>
- [6] Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323(6088), 533-536. Преузето 27.10.2025. са: <https://www.nature.com/articles/323533a0>
- [7] Ralf C. Staudemeyer, Eric Rothstein Morris (2019). Understanding LSTM -- a tutorial into Long Short-Term Memory Recurrent Neural Networks. Преузето 27.10.2025.

Кратка биографија:



Марко Василић рођен је 15. септембра 1998. године у Бањој Луци. Мастер рад на Факултету техничких наука у Новом Саду из области Електротехнике и рачунарства одбранио је 2025. године