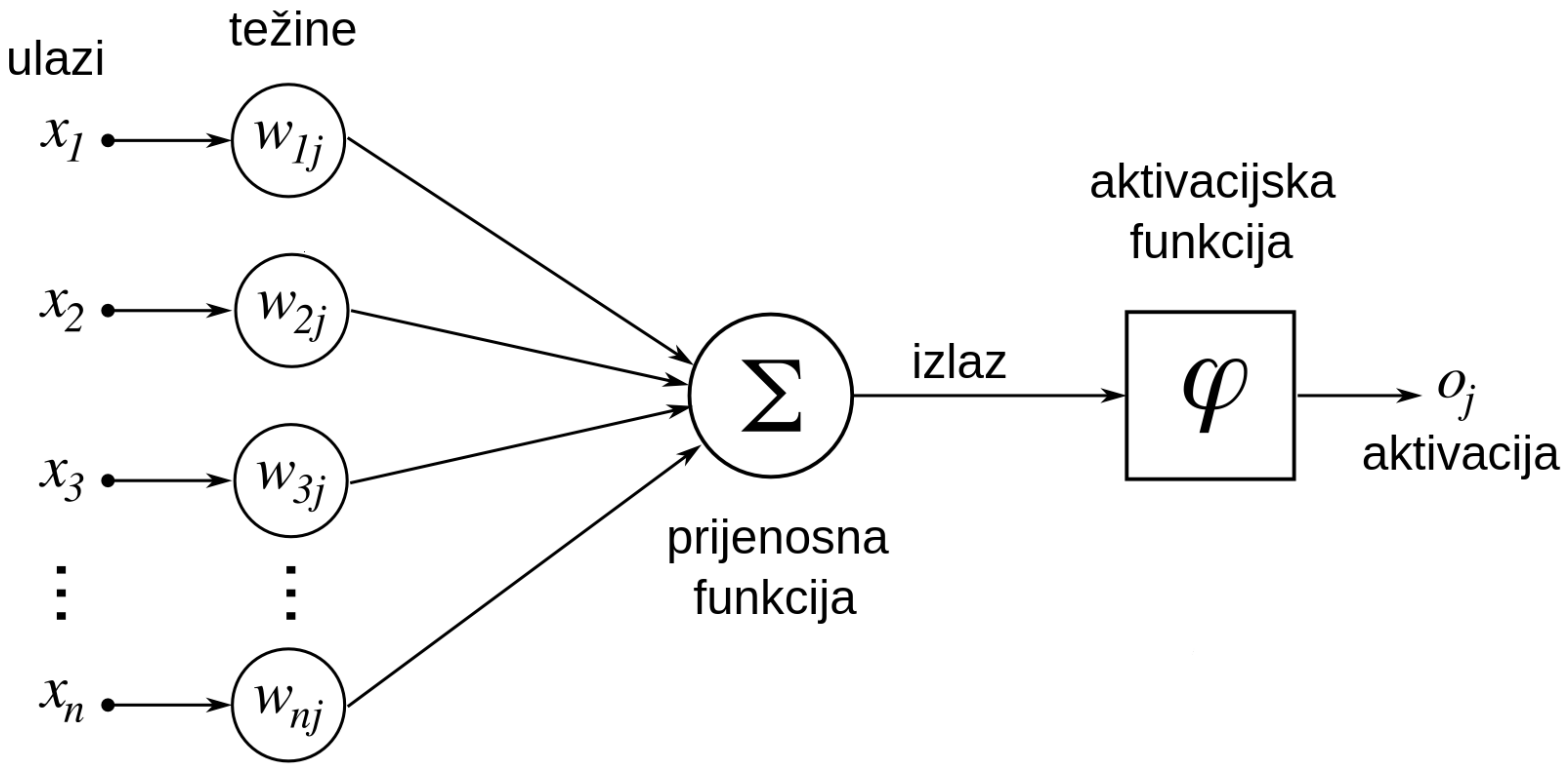
BIBLIOTEKA ZA SPECIFIKACIJU I UČENJE NEURONSKIH MREŽA

Damir Numić-Meša

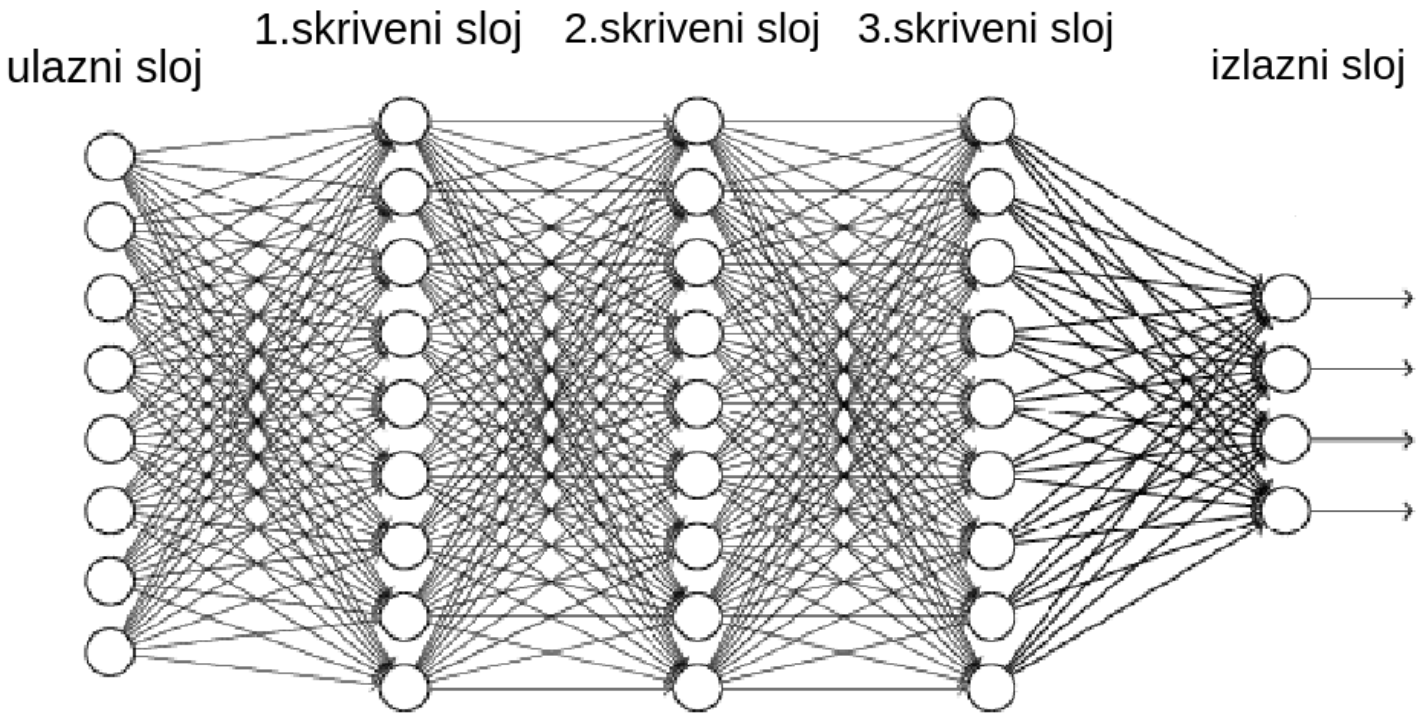
POGLAVLJA

NEURON

- osnovna gradivna jedinica neuronske mreže
- matematička funkcija
- ulazima pridruženi koeficijenti - težine

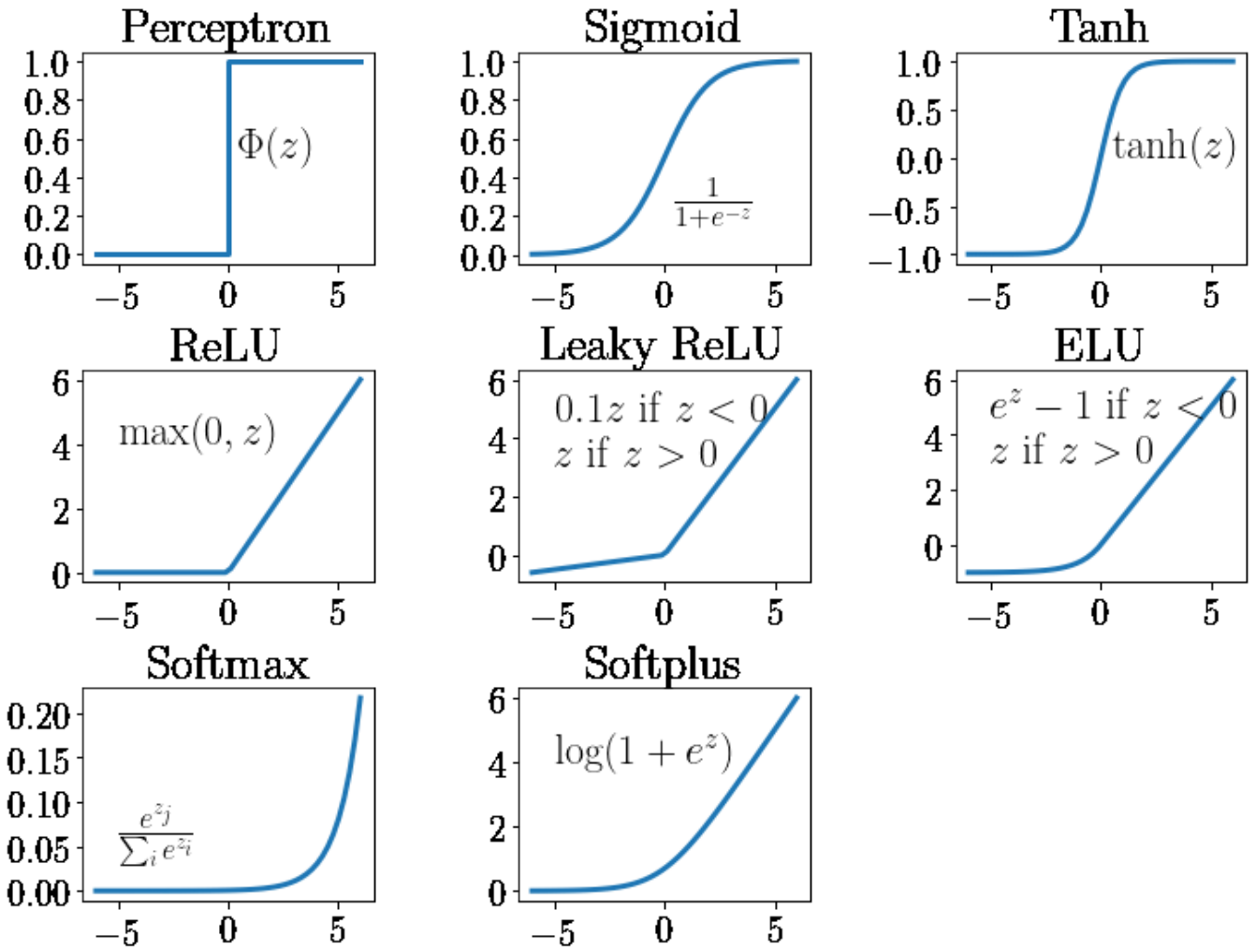


POVEZIVANJE VIŠE NEURONA



AKTIVACIJSKE FUNKCIJE

- transformacija izlaza neurona
- uvođenje nelinearnosti -> aproksimacija nelinearnih funkcija

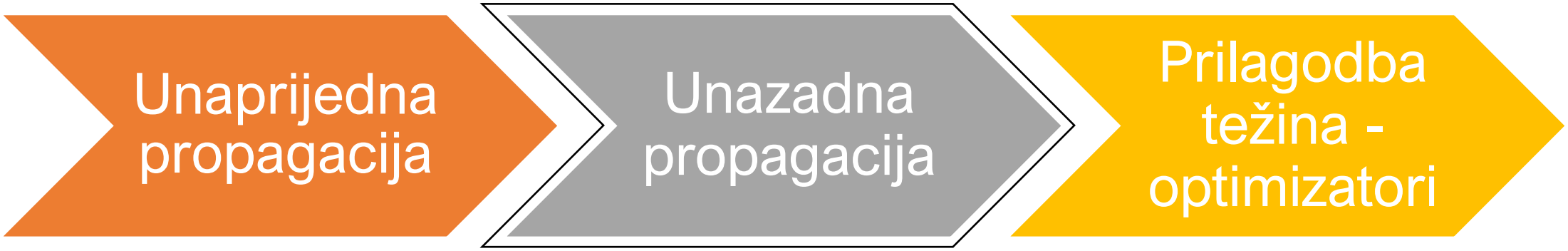


KAKO NEURONSKA MREŽA UČI?

KAKO NEURONSKA MREŽA UČI?

- ulazni podaci transformirani od neurona ulaznog sloja do neurona izlaznog sloja

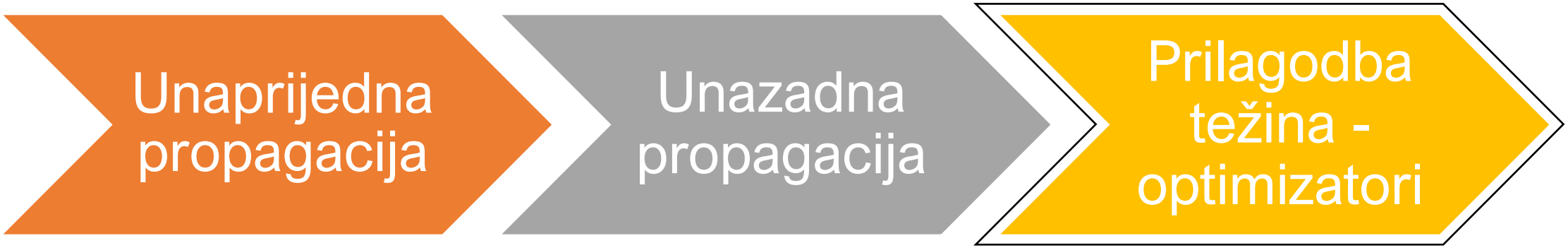
KAKO NEURONSKA MREŽA UČI?



- ulazni podaci transformirani od neurona ulaznog sloja do neurona izlaznog sloja

- funkcija pogreške
- delta – utjecaj svakog neurona na pogrešku
- gradijent

KAKO NEURONSKA MREŽA UČI?



- ulazni podaci transformirani od neurona ulaznog sloja do neurona izlaznog sloja

- funkcija pogreške
- delta – utjecaj svakog neurona na pogrešku
- gradijent

- prilagodba matrice težina gradijentom

OPTIMIZATORI

- algoritmi prilagodbe matrica težina
- SGD i ADAM optimizator

SGD

- problematična nomenklatura
- jednostavan algoritam
- α – stopa učenja

$$w^{l}_{(t+1)} = w_t - \alpha \frac{\partial L}{\partial w^{(l)}}$$

ADAM

- spoj više algoritama
- estimacija prvog i drugog momenta gradijenta
- α – stopa učenja
- β_1, β_2 – decay rate

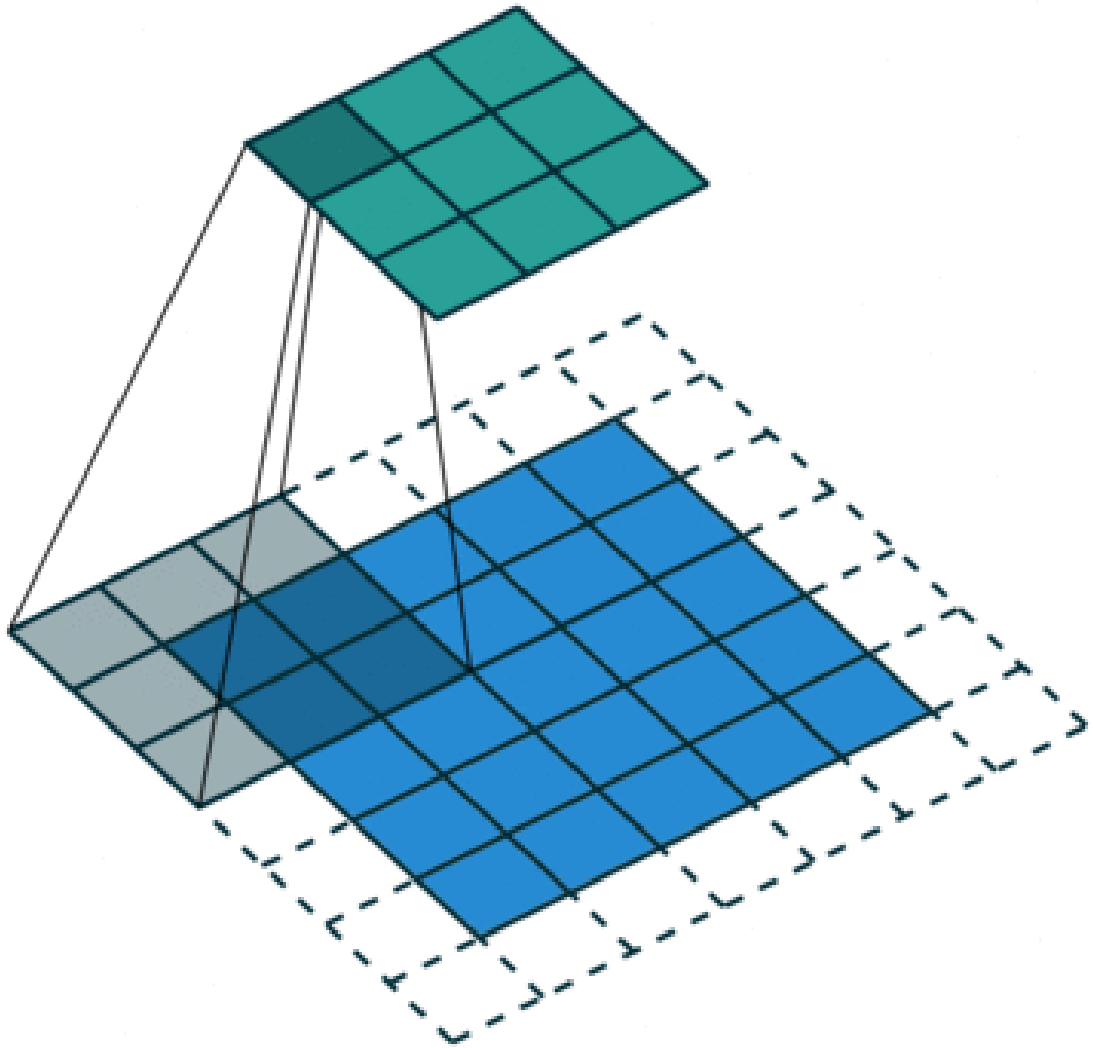
1. $$m_{t+1}^{(l)} = \beta_1 * m_t^{(l)} + (1 - \beta_1) * \left(\frac{\partial L}{\partial w_t^{(l)}} \right)$$

2. $$v_{t+1}^{(l)} = \beta_2 * v_t^{(l)} + (1 - \beta_2) * \left(\frac{\partial L}{\partial w_t^{(l)}} \right)^2$$

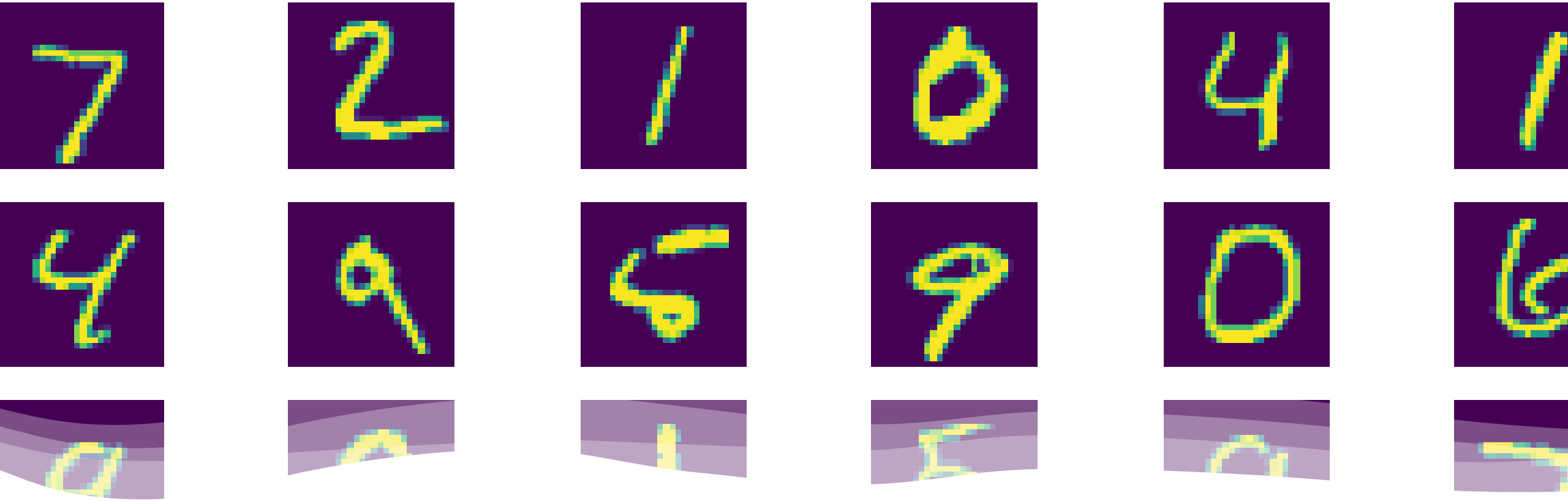
3. $$w_{(t+1)}^{(l)} = w_{(t)}^{(l)} - \alpha * \frac{\hat{m}_{t+1}^{(l)}}{\sqrt{\hat{v}_{t+1}^{(l)} + \epsilon}}$$

KONVOLUCIJA

- filter veličine k x w
- izdvajanje značajki slike
- redukcija dimenzionalnosti



KLASIFIKACIJSKI PROBLEMI

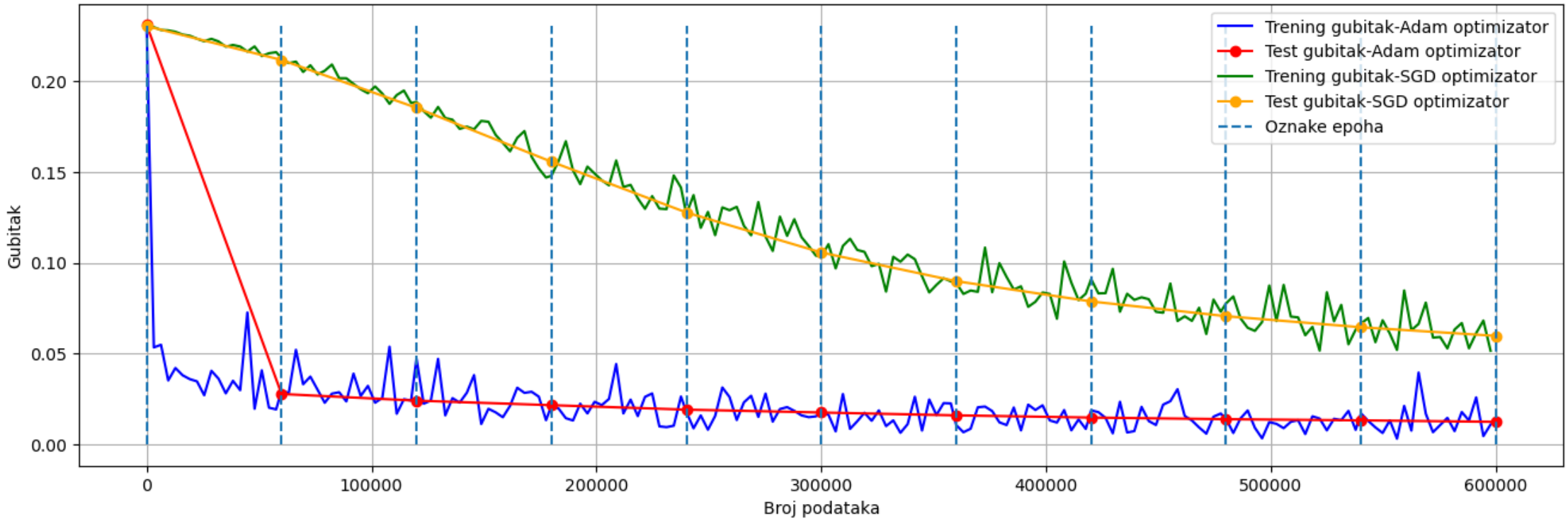


MNIST

- slike 28 x 28 piksela
- 10 klasa
- podjela 60000/10000 za trening/testiranje
- treniranje na 10 epoha

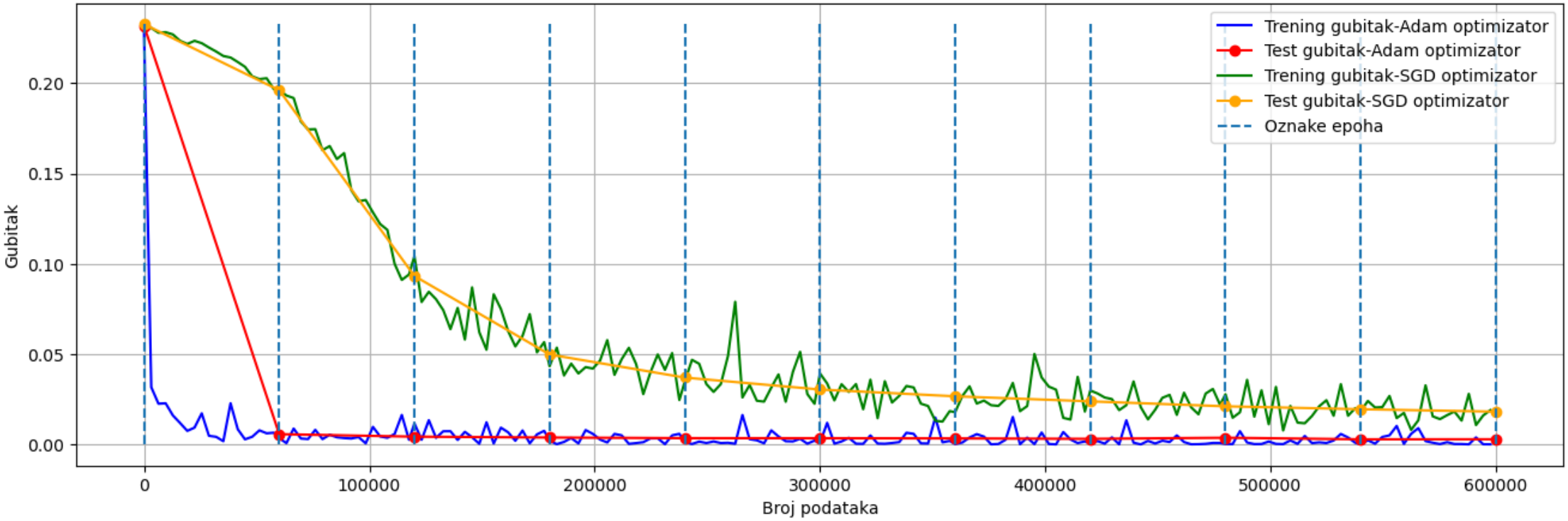
1.ARHITEKTURA

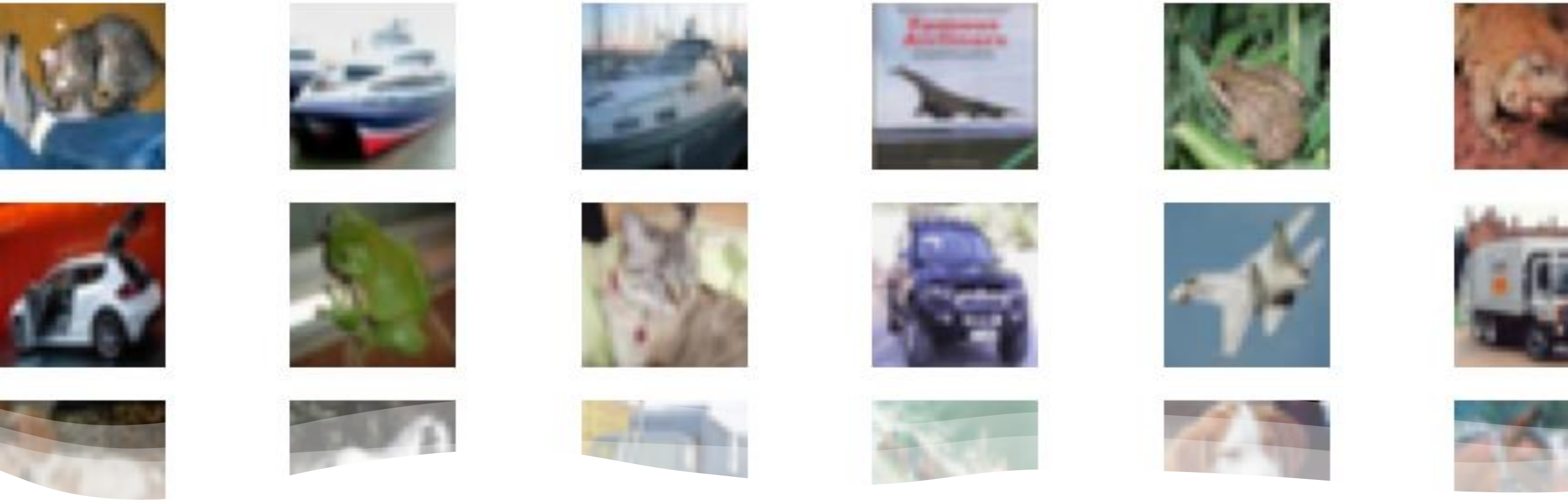
- 784 x 50 x 10
- potpuno povezana mreža
- 96,36%



2.ARHITEKTURA

- konvolucijska neuronska mreža od 3 sloja
- maxpooling slojevi
- veća točnost na skupu za testiranje (99.15% vs 96.36%) u usporedbi sa potpuno povezanom arhitekturom



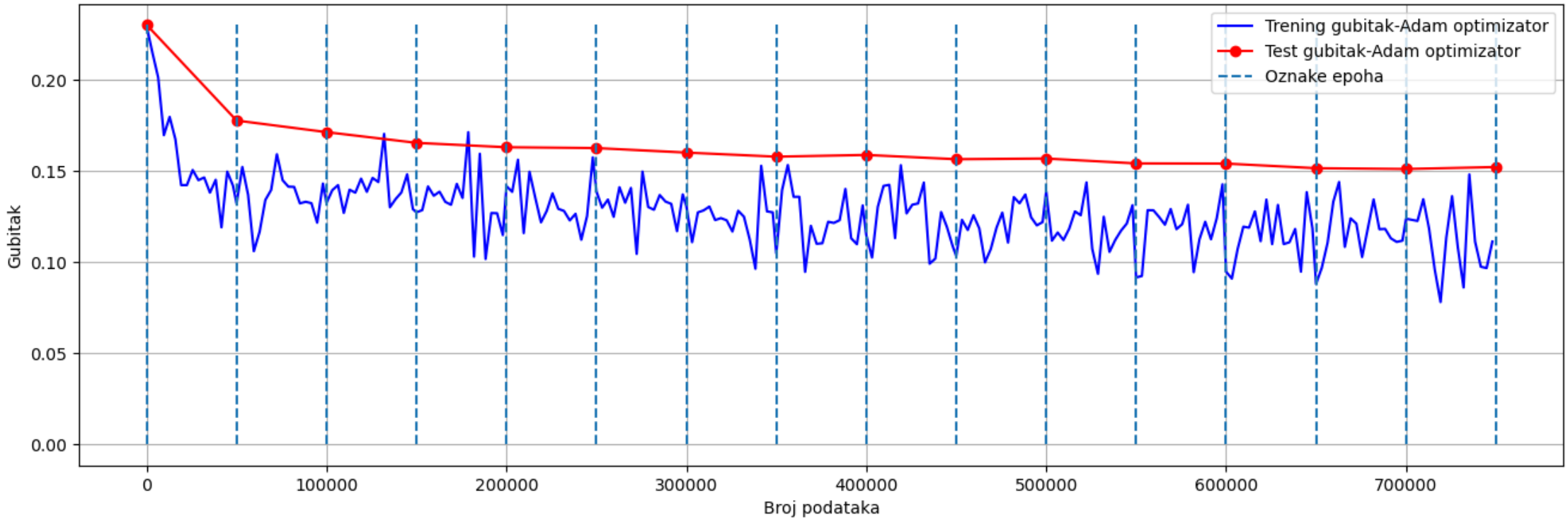


CIFAR10

- slike 32 x 32 piksela
- 10 klasa (zrakoplov, automobil, ptica, mačka...)
- podjela 50000/10000 za trening/testiranje
- treniranje na 15 epoha

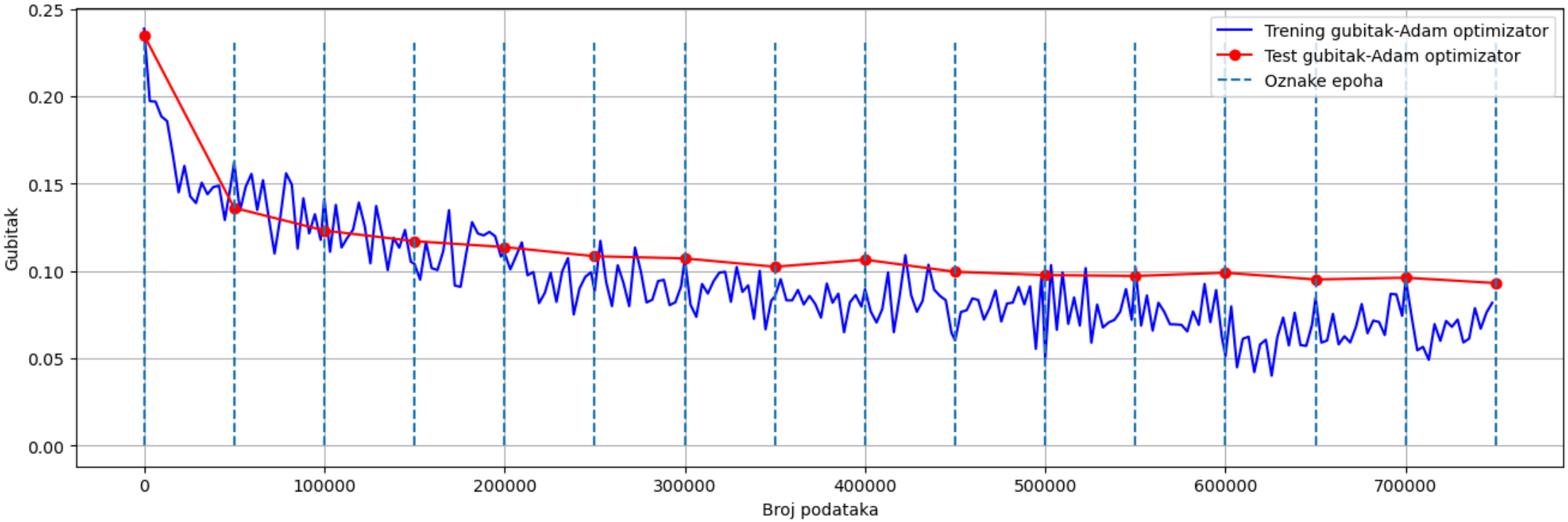
1.ARHITEKTURA

- 3072 x 256 x 10
- potpuno povezana mreža
- 46,31% klasifikacijske točnosti



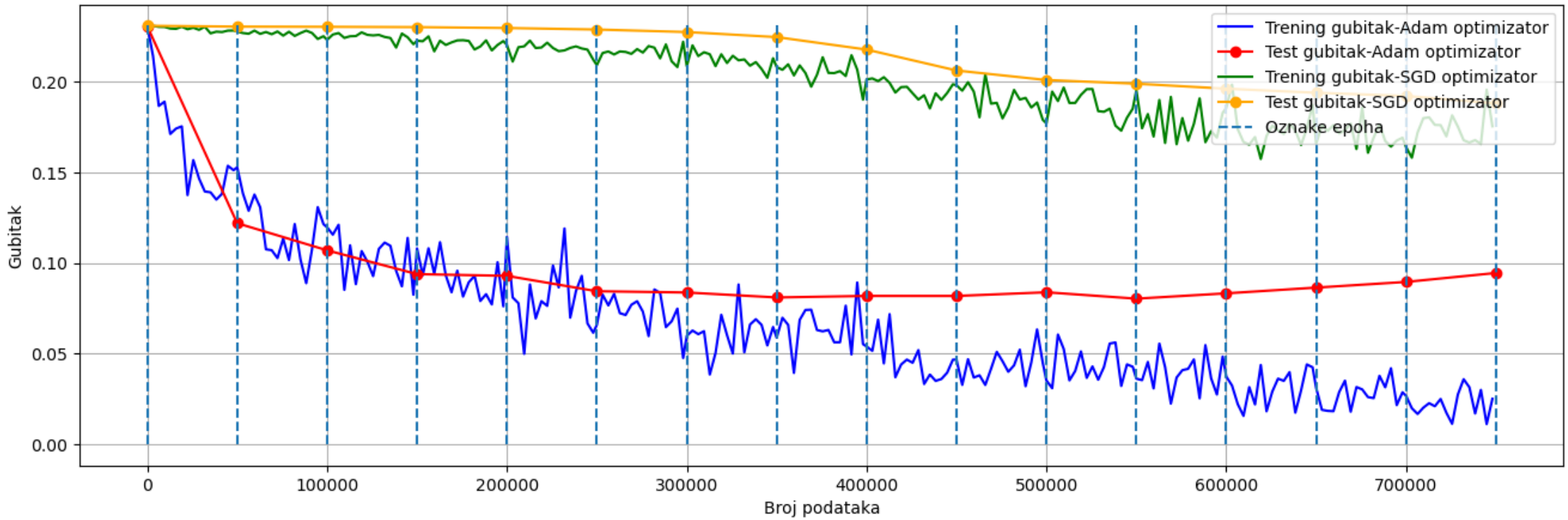
2.ARHITEKTURA

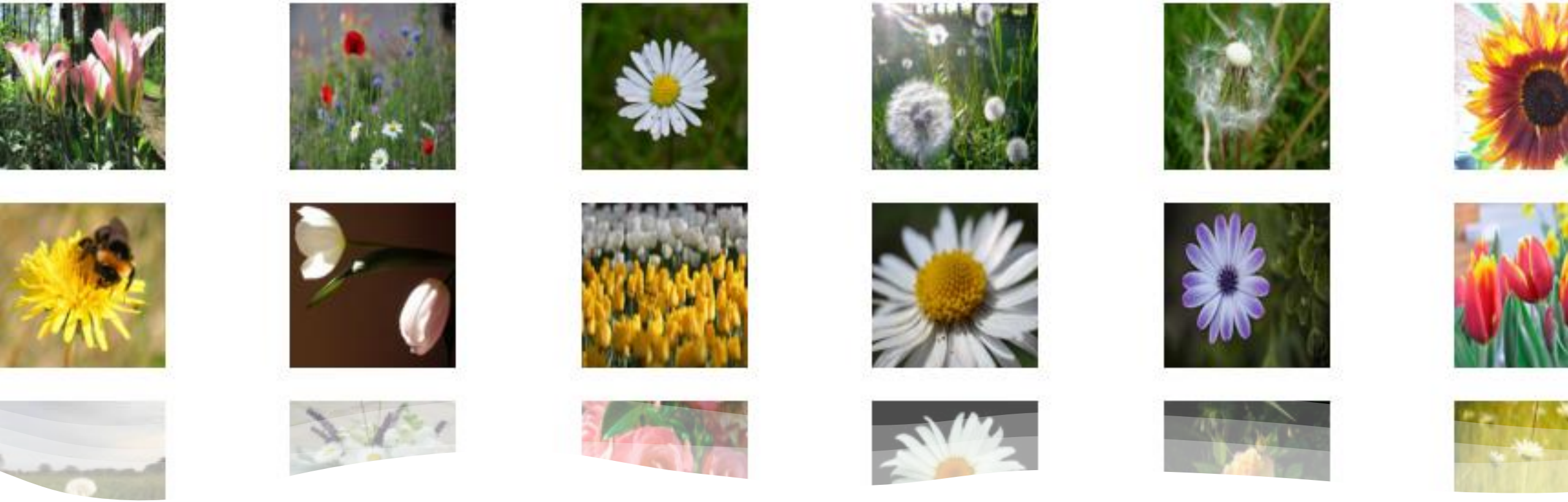
- konvolucijska neuronska mreža od 3 konvolucijska sloja
- maxpooling slojevi
- 67,75% klasifikacijske točnosti



3.ARHITEKTURA

- povećana kompleksnost konvolucijske mreže - 6 konvolucijskih slojeva + maxpooling slojevi
- 74,45% točnosti na skupu za testiranje
- problem prenaučenosti



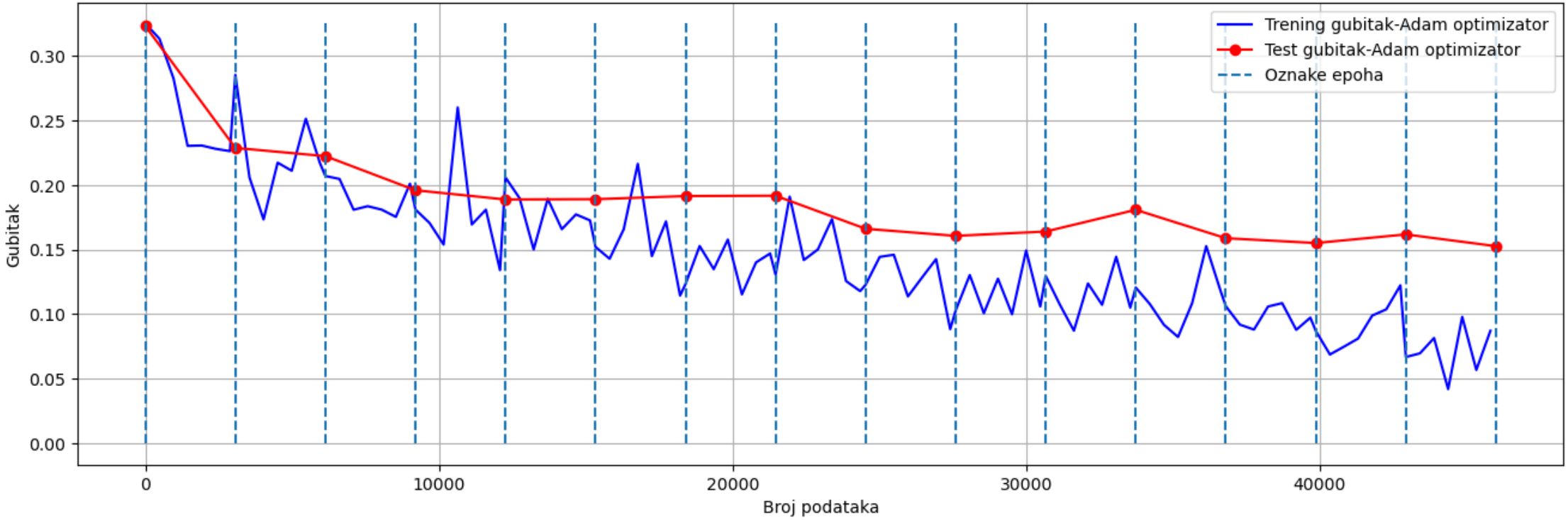


FLOWERS

- slike 128 x 128 piksela
- 5 klasa (kamilica, tulipan, ruža...)
- podjela 3067/1250 za trening/testiranje
- treniranje na 15 epoha

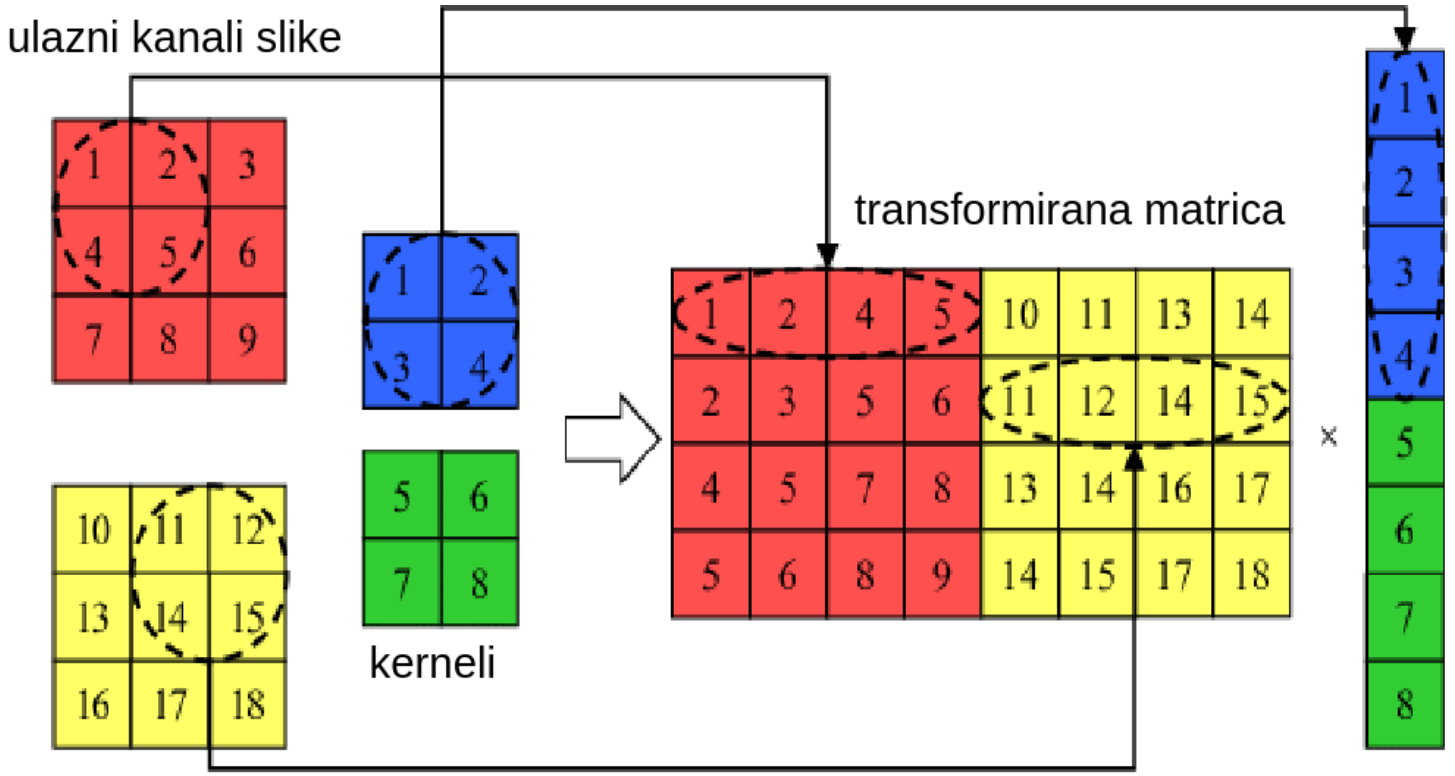
ARHITEKTURA

- 4 konvolucijska sloja + maxpooling slojevi
- 72,96% točnosti na skupu za testiranje



IMPLEMENTACIJSKE POJEDINOSTI

- Python + CuPy
- implementacija konvolucije – im2col i col2im, značajno brže nego "klasični" pristup
- verifikacija algoritma unazadne propagacije numeričkom aproksimacijom gradijenta



ZAKLJUČAK

- shvaćanje matematičke pozadine neuronskih mreža i načina implementacije
- potencijalna poboljšanja i proširenje trenutne izvedbe biblioteke