Neuralne mreže

Projektni zadatak

2024/25

Andrej Praizović

0300/22

Lea Irt

0371/22

Zadatak 1

Projektovanje potpuno povezane neuralne mreže

Potrebno je projektovati višeslojnu potpuno povezanu neuralnu mrežu za klasifikaciju odbiraka u više klasa. Skup podataka je: „**Occupancy.csv**“.

Potrebno je opisati problem koji se rešava, što podrazumeva navodenje osnovnih karakteristika skupa podataka: broj uzoraka, broj obeležja (atributa), broj klasa i priroda klasifikacionog zadatka.

Zatim je neophodno konstruisati i obučiti višeslojnu potpuno povezanu (feedforward) neuronsku mrežu za rešavanje datog problema. Treba obrazložiti izbor: kriterijumske funkcije, funkcije aktivacija neurona, kao i metode optimizacije korišćene za minimizaciju funkcije greške. Za svaki od ovih izbora potrebno je kratko objasniti teorijsku osnovu i opravdati njihovu primenu u konkretnom problemu.

Primenom unakrsne validacije pronaći optimalan set hiperparametara po izboru. Potrebno je izabrati najmanje tri hiperparametra, opisati njihove uloge, i navesti koje su vrednosti testirane tokom validacije.

Od rezultata prikazati i komentarisati:

• histogram raspodele odbiraka po klasama,

• grafik promena performansi tokom epoha treniranja (za trening i validacioni skup),

• vrednosti performansi klasifikacije, kao što su: tačnost, preciznost, osetljivost, f1-skor,

• matricu konfuzije na trening i test skupu.

Problem

U pitanju je predikcija da li se u prosotriji nalazi neko ili je prostorija prazna, na osnovu izmerenih uslova u prostoriji (temperatura, vlažnost vazduha, svetlost, koncentracija CO_2,...).

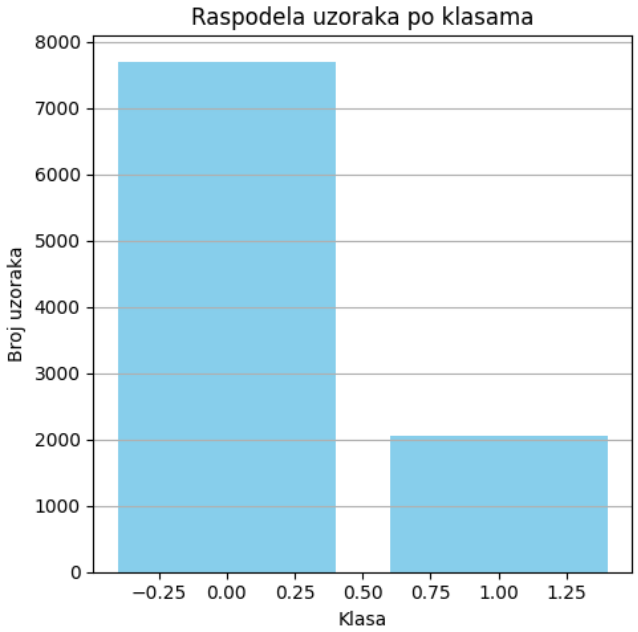
Osnovne karakteristike skupa podataka

Broj uzoraka: 9752

Broj obeležja: 5 (Temperature, Humidity, Light, CO_2, HumidityRatio)

Broj klasa: 2 (K0 - prazna prostorija, K1 - neko se nalazi u prostoriji)

Priroda klasifikacionog zadatka: binarna klasifikacija - zauzetost prostorije (0 - nezauzeta, 1-zauzeta), klase su neuravnotežene (K0 - 7703 uzoraka, K1 - 2049 uzoraka)



Slika1. Histogram raspodele odbiraka po klasama

Izbori i obrazloženja

Kriterijumska funkcija: **`binary cross-entropy`**

Ovo je funkcija gubitka koja se koristi za probleme binarne klasifikacije. Meri koliko mreža greši tako što poredi predvidjene vrednosti izlaza sa stvarnim izlazima (0 i 1). Što je njena vrednost manja, model je bolji. Izabrali smo je jer je standard za binarnu klasifikaciju i daje stabilne gradijente.

Funkcije aktivacije: izlazni sloj - **`unipolarni sigmoid`**, skriveni slojevi - **`relu`**

Unipolarni sigmoid je funkcija aktivacije koja vraća vrednosti od 0 do 1, što je pogodno za ovaj problem i dobra je interpretacija načina funkcionisanja pravog neurona. Relu je funkcija aktivacije koja omogućava brže učenje, jer gradijenti ne eksplodiraju/nestaju (za pozitivne vrednosti koje posmatramo u ovom problemu). Efikasna je za izvršavanje i u praksi brzo konvergira ka rešenju.

Metode optimizacije: **`adam`**

Ovo je najpopularniji optimizator, jer kombinuje momentum i rmsprop, uzimajući najbolje karakteristike oba.

Dodatna poboljšanja

Zbog neuravnoteženosti klasa, bilo je potrebno balansirati podatke. Najpre smo pokušali oversampling metodom, konkretno smo koristili smote verziju, pogodna je zato što kombinuje postojeće podatke i na taj način pravi nove, različite od postojećih (ovaj deo koda

je zakomentarisan). Potom smo pokušali metodom dodavanja težina klasama, što je dalo bolje rezultate na konkretnom primeru.

```
Težine klasa su: K0 - 0.6297941057731127, K1 - 2.426127527216174
```

Slika2. Dobijene težine klasa K0 i K1

Može se uočiti da je priroda obeležja raznolika. *Temperature* je u °C i uzima vrednosti oko 21, a obeležja poput *HumidityRatio* su reda veličine 10^{-3}. Kako bismo sva obeležja sveli na isti opseg i kako bi podjednako doprinosila obučavanju, koristimo skaliranje. Mi smo se opredelili za minmax scaling.

	count	mean	std	min	25%	50%	75%	max
Temperature	6240.0	21.005994	1.028446	19.500000	20.290000	20.790000	21.55000	24.390000
Humidity	6240.0	29.853821	3.955225	21.890000	26.550000	30.166667	32.64500	39.500000
Light	6240.0	122.465254	209.239126	0.000000	0.000000	0.000000	192.81250	1581.000000
CO2	6240.0	754.550656	297.285178	484.666667	543.000000	641.625000	833.50000	1879.500000
HumidityRatio	6240.0	0.004584	0.000529	0.003279	0.004188	0.004584	0.00499	0.005769

Slika3. Statistika obeležja pre skaliranja

	count	mean	std	min	25%	50%	75%	max
Temperature	6240.0	0.307974	0.210316	0.0	0.161554	0.263804	0.419223	1.0
Humidity	6240.0	0.452233	0.224601	0.0	0.264622	0.469998	0.610733	1.0
Light	6240.0	0.077461	0.132346	0.0	0.000000	0.000000	0.121956	1.0
CO2	6240.0	0.193488	0.213133	0.0	0.041821	0.112528	0.250090	1.0
HumidityRatio	6240.0	0.524197	0.212575	0.0	0.365038	0.524324	0.687457	1.0

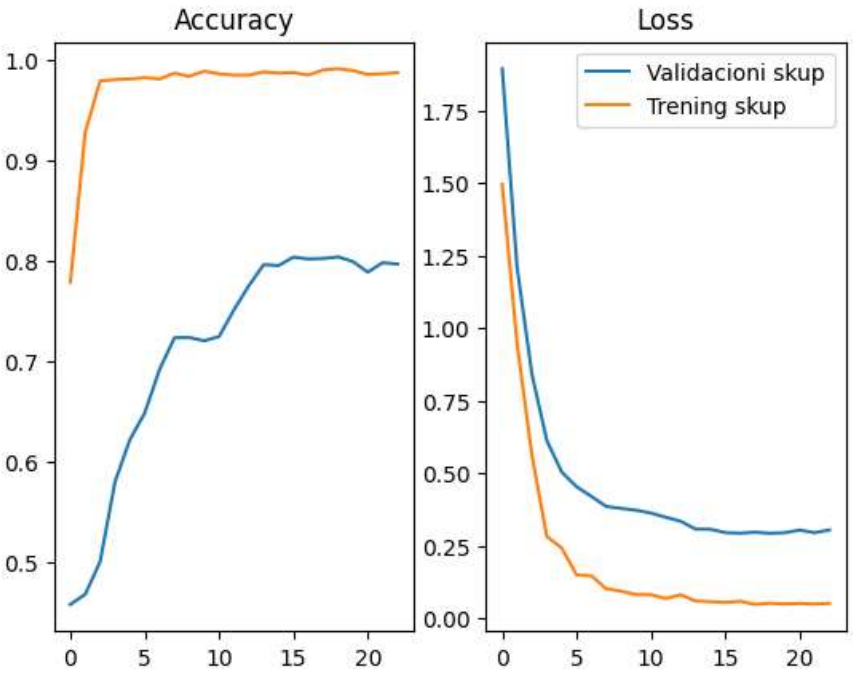
Slika4. Statistika obeležja nakon skaliranja

Među skrivenim slojevima uveli smo batch normalizaciju. Ona normalizuje aktivacije po batch-u (koristi srednju vrednost i varijansu svakog obeležja ponaosob), smanjuje oscilacije ulaza za svaki sloj i time ubrzava i stabilizuje konvergenciju, pa se može koristiti veća konstanta obučavanja.

Koristili smo l2 regularizaciju, jer penalizuje velike težine i „vraća“ ih ka nuli. To sprečava da model nauči šum iz podataka i održava težine malim.

Kako bismo sprečili preobučavanje i zaustavili učenje na vreme uveli smo early stopping. On prekida trening kada se validacioni loss ne poboljšava zadati broj epoha i vraća najbolje težine, što daje bolje finalne performanse i štedi računarske resurse.

Metodu dropout smo koristili da nasumično isključuje neurone tokom treninga. Zahvaljujući njoj mreži nije dozvoljeno da „uči napamet“ i oslanja se na prečice. Ovo nam daje bolju generalizaciju na test podacima koje mreža nije videla pre i korisno je kad imamo ograničen broj uzoraka.



Slika5. Grafik promena performansi tokom epoha treniranja (za trening i validacioni skup)

Hiperparametri i tjuning

Optimalan broj neurona u prvom skrivenom sloju - **`units`**

Broj neurona ne sme biti ni previše velik, jer dovodi do prekomernog prilagođavanja šumu, ali ni previše mali, jer model onda nema mogućnost učenja složenih obrazaca iz podataka.

Testirane vrednosti: `min = 5, max = 15`

Optimalna funkcija aktivacije u prvom skrivenom sloju - **`activation`**

Svrha funkcije aktivacije jeste da omogući modelovanje nelinearnih funkcija i time čini mrežu sposobnom da uči složene obrasce. Promena izbora funkcije aktivacije menja dinamiku gradijenta, stabilnost i brzinu konvergencije. Lošiji izbor funkcije aktivacije može dovesti do toga da je neophodna manja konstanta obučavanja, pa je učenje sporije.

Testirane vrednosti: `['sigmoid', 'relu', 'tanh']`

Optimalan koeficijent regularizacije u drugom skrivenom sloju - **`reg`**

Jačina L2 regularizacije predstavlja to koliko jako „vučemo“ velike težine ka nuli. Ako je prevelika - radićemo sa dosta malim težinama, a ako je premala - nema efekta.

Testirane vrednosti: `0.001, 0.5, 0.005`

Optimalan dropout rate - **`drop`**

Dropout nasumično isključuje neurone tokom treninga sa verovatnoćom drop. Ako je drop previše mali - nema efekta, a ako je previše veliki - model poduči, jer gubi previše kapaciteta.

Testirane vrednosti: `0, 0.8, 0.1`

Optimalna konstanta obučavanja - **`learning_rate`**

Konstanta obučavanja kontroliše veličinu koraka koje optimizator pravi pri ažuriranju težina. Ako je prevelika - dovodi do oscilacija, a ako je premala - čini učenje veoma sporim i lako zaglavi u lokalnim minimumima.

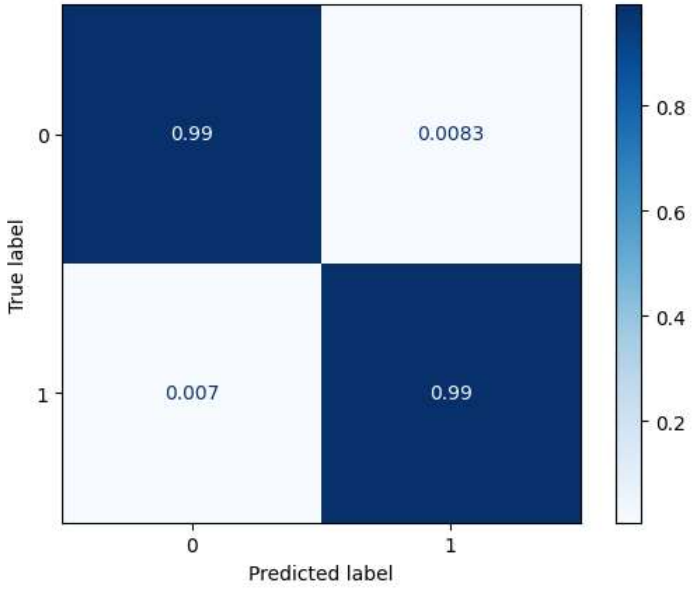
Testirane vrednosti: `1e-5, 1e-2, 1e-4`

```
print('Optimalan broj neurona u prvom skrivenom sloju: ', best_hyperparam['units'])
print('Optimalna funkcija aktivacije u prvom skrivenom sloju: ', best_hyperparam['activation'])
print('Optimalan koeficijent regularizacije u drugom skrivenom sloju: ', best_hyperparam['reg'])
print('Optimalan dropout rate: ', best_hyperparam['drop'])
print('Optimalna konstanta obučavanja: ', best_hyperparam['learning_rate'])
```

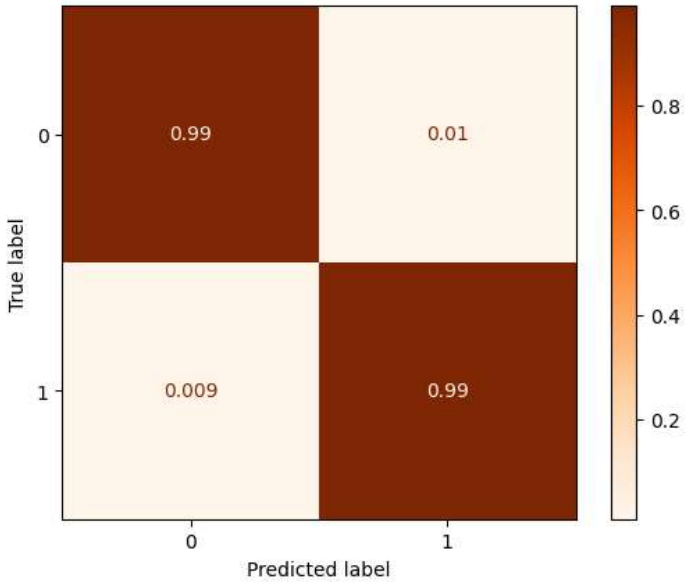
```
[360] ✓ 0.0s

Optimalan broj neurona u prvom skrivenom sloju:  5
Optimalna funkcija aktivacije u prvom skrivenom sloju:  relu
Optimalan koeficijent regularizacije u drugom skrivenom sloju:  0.186
Optimalan dropout rate:  0.7000000000000001
Optimalna konstanta obučavanja:  0.0013100000000000002
```

Slika7. Izračunati najbolji hiperparametri



Slika8. Matrica konfuzije na trening skupu



Slika9. Matrica konfuzije na test skupu

```
Tacnost na test skupu iznosi: 98.82111737570477%
Preciznost na test skupu iznosi: 98.85699984926455%
Osetljivost na test skupu iznosi: 98.82111737570477%
F1-skor: 98.82879305122%
```

Slika10. Vrednosti performansi klasifikacije (tačnost, preciznost, osetljivost, f1-skor) na test skupu

```
Klasa '0':
  Tačnost: 0.990
  Preciznost: 0.993
  Osetljivost: 0.987
  F1-skor: 0.990

Klasa '1':
  Tačnost: 0.990
  Preciznost: 0.987
  Osetljivost: 0.993
  F1-skor: 0.990
```

Slika11. Vrednosti performansi klasifikacije (tačnost, preciznost, osetljivost, f1-skor) za pojedinačne klase

Zadatak 2

Projektovanje potpuno konvolucione neuralne mreže

Potrebno je projektovati konvolucionu neuralnu mrežu (CNN) za klasifikaciju slika u više klasa. Skup podataka koji će se koristiti studenti pronalaze samostalno, pri čemu je važno da sadrži dovoljno slika rasporedenih u više jasno definisanih klasa.

Potrebno je opisati prirodu problema koji se rešava (tip ulaznih podataka, broj i naziv klasa) i prikazati grafikon koji prikazuje broj uzoraka po klasama.

Zatim je potrebno izvršiti predprocesiranje podataka (npr. skaliranje, normalizaciju, augmentaciju itd.). Treba navesti koje su transformacije primenjene, sa obrazloženjem njihove svrhe i uticaja na model.

Na osnovu obradenih podataka, potrebno je formirati i obučiti konvolucionu mrežu za klasifikaciju. Pri tome je potrebno obrazložiti izbor: kriterijumske funkcije, funkcije aktivacija neurona, kao i metode optimizacije korišćene za minimizaciju funkcije greške. Na kraju, potrebno je prikazati arhitekturu razvijenog modela (npr. u koristeći model.summary()), kao i navesti ukupan broj parametara u modelu.

Od rezultata prikazati i komentarisati:

• po jedan primerak iz svake klase,

• grafik promena performansi tokom epoha treniranja (za trening i validacioni skup),

• vrednosti performansi klasifikacije, kao što su: tačnost, preciznost, osetljivost, f1-skor,

• matricu konfuzije na trening i test skupu,

• primere dobro i loše klasifikovanih slika iz skupa podataka.

Problem

Raspoznavanje autora zadate slike. Model se obučava da razlikuje dela ovih 5 slikara: Endi Vorhol, Pablo Pikaso, Rembrant, Salvador Dali, Vinsent van Gog.

Podaci su preuzeti sa sajta: https://www.kaggle.com/datasets/ikarus777/best-artworks-of-all-time

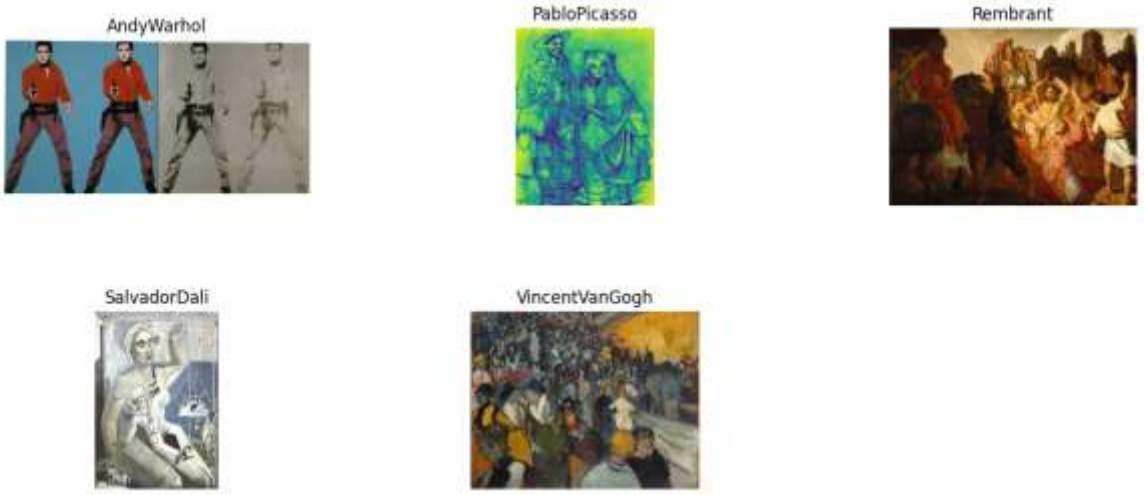
Osnovne karakteristike skupa podataka

Broj uzoraka: 880 slika

Broj klasa: 5 ('Andy Warhol', 'Pablo Picasso', 'Rembrant', 'Salvador Dali', 'Vincent Van Gogh']

Slika12. Nasumičnih 10 uzoraka

Slika13. Raspored uzoraka po klasama



Slika14. Po jedan primerak iz svake klase

Predprocesiranje podataka

Skaliranje

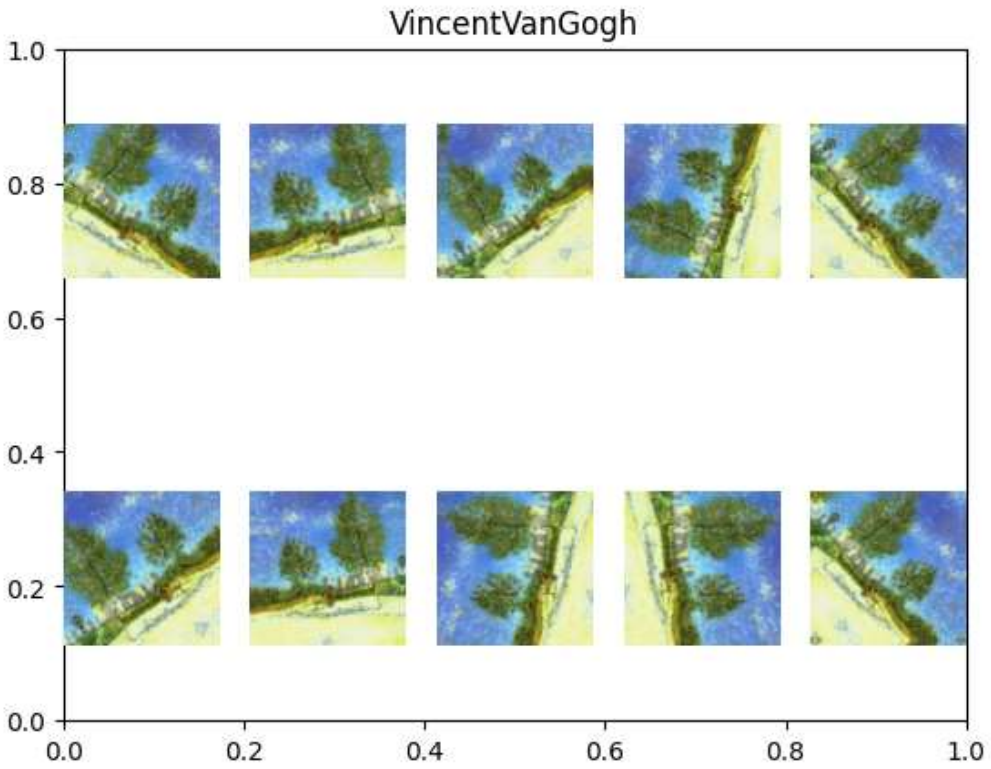
Sve slike sa njihovih originalnih dimenzija skaliramo na 128x128x3. To omogućava konzistentan ulaz u mrežu, smanjuje varijabilnost dimenzija koja usporava optimizator i omogućava efikasnu batch obradu, što obučavanje čini bržim.

Normalizacija

Koristili smo batch normalizaciju koja normalizuje aktivacije unutar svakog batch-a (i uči skaliranje i pomak). To smanjuje oscilovanje distribucije aktivacija između epoha, stabilizuje gradijente, samim tim možemo koristiti veću konstantu obučavanja, pa je i učenje stabilnije i brže.

Augmentacija

Ona podrazumeva dodavanje modifikovanih ili sintetički generisanih slika, kako bi model bolje generalizovao. U našem kodu to je implementirano tako što se za svaku sliku iz ulaznih podataka primenjuju rotacije, zoom i flip.



Slika15. Vizuelni prikaz N=10 augmentacija jedne Van Gogove slike

Izbori i obrazloženja

Kriterijumska funkcija: **`SparseCategoricalCrossentropy`**

Ovo je funkcija gubitka koja se koristi za probleme višeklasne klasifikacije. Odgovara konkretnom problemu, jer se labele klasa mogu predstaviti u obliku celih brojeva (['Andy Warhol', 'Pablo Picasso', 'Rembrant', 'Salvador Dali', 'Vincent Van Gogh'] → [0 - 4]). Minimizacija ove funkcije vodi ka boljim verovatnoćama po klasama.

Funkcije aktivacije: konvolucioni slojevi i skriveni sloj - **`relu`,** izlazni sloje - **`softmax`**

Relu koristimo u konvolucionim i skrivnim Dense slojevima, jer je to jednostavna funkcija aktivacije, u praksi brzo konvergira ka rešenju, izbegava zasićenje za pozitivne ulaze (nema saturacije). U izlaznom sloju korisnimo softmax koji pretvara realne izlaze u verovatnoće.

Metode optimizacije: **`adam`**

Ovo je najpopularniji optimizator, jer kombinuje momentum i rmsprop, uzimajući najbolje karakteristike oba. U praksi brzo daje stabilne rezultate za konvolucione mreže.

Dodatna poboljšanja

Nakon poslednjeg konvolucionog sloja koristimo dropout regularizaciju koja sprečava da dodje do preobučavanja tako što tokom obučavanja „gasi“ nasumične neurone.

Kako bismo dodatno sprečili preobučavanje, skratili vreme učenja i dobili bolju generalizaciju i bolje finalne performanse modela, uveli smo i early stopping.

Model: "sequential_33"

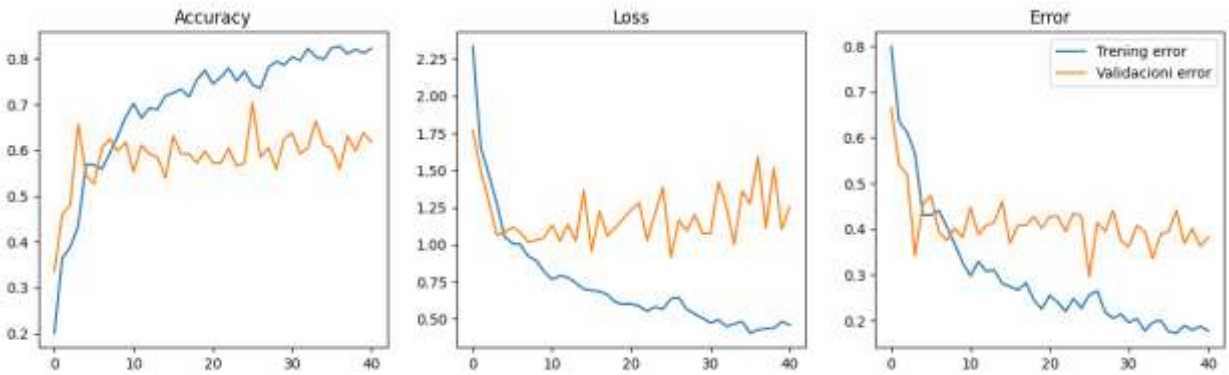
Layer (type)	Output Shape	Param #
sequential_32 (Sequential)	(None, 128, 128, 3)	0
rescaling_20 (Rescaling)	(None, 128, 128, 3)	0
conv2d_63 (Conv2D)	(None, 128, 128, 16)	448
max_pooling2d_43 (MaxPooling2D)	(None, 64, 64, 16)	0
conv2d_64 (Conv2D)	(None, 64, 64, 32)	4,640
max_pooling2d_44 (MaxPooling2D)	(None, 32, 32, 32)	0
conv2d_65 (Conv2D)	(None, 32, 32, 64)	18,496
dropout_20 (Dropout)	(None, 32, 32, 64)	0
flatten_20 (Flatten)	(None, 65536)	0
dense_40 (Dense)	(None, 128)	8,388,736
dense_41 (Dense)	(None, 5)	645

Total params: 8,412,965 (32.09 MB)

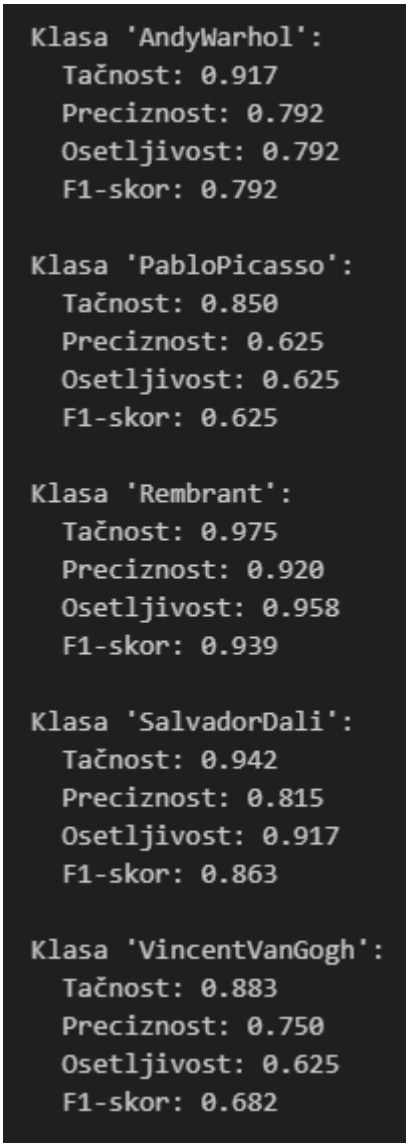
Trainable params: 8,412,965 (32.09 MB)

Non-trainable params: 0 (0.00 B)

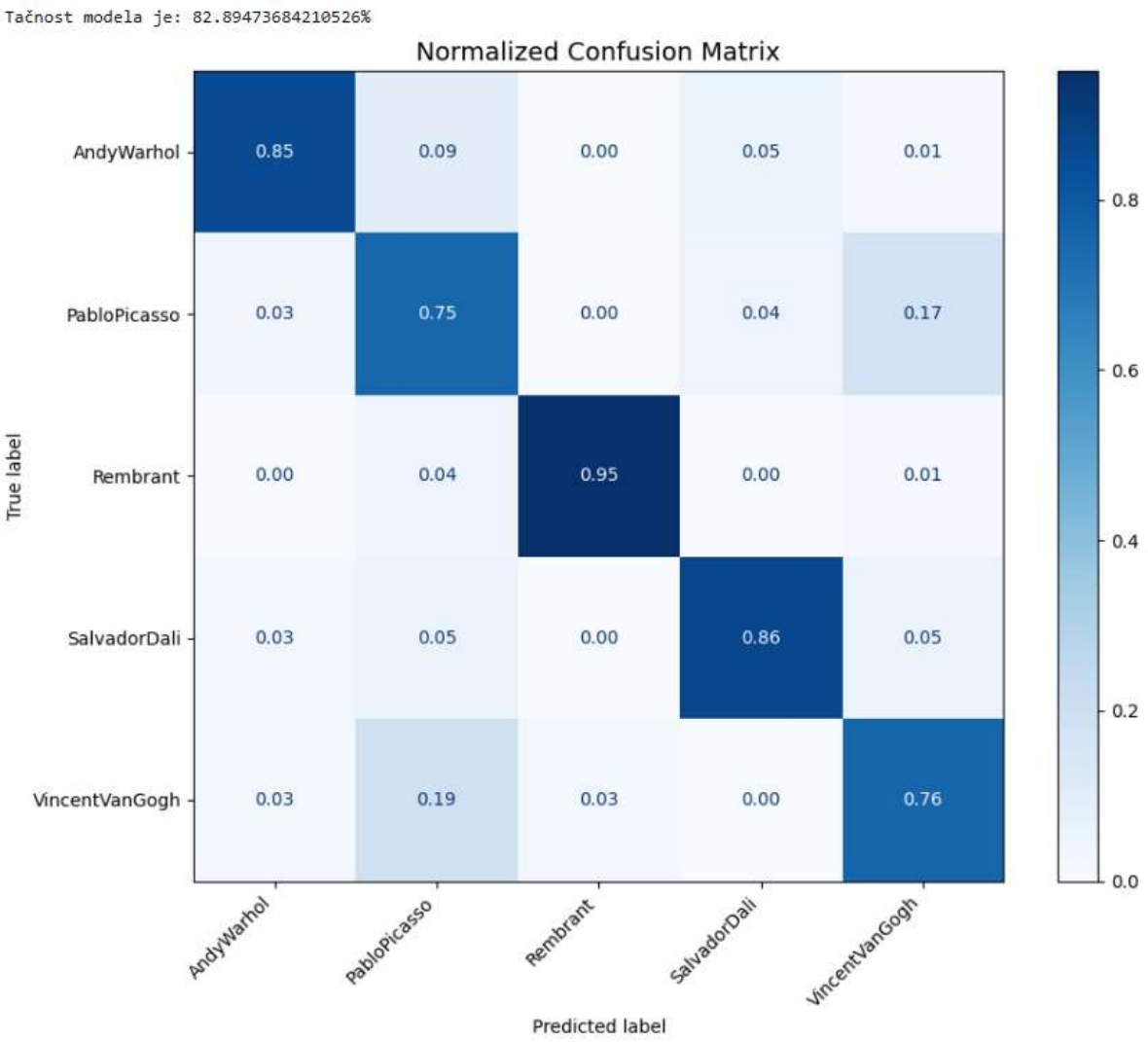
Slika16. Prikaz arhitekture razvijenog modela i ukupan broj parametara u modelu



Slika17. Grafik promena performansi tokom epoha treniranja (za trening i validacioni skup)

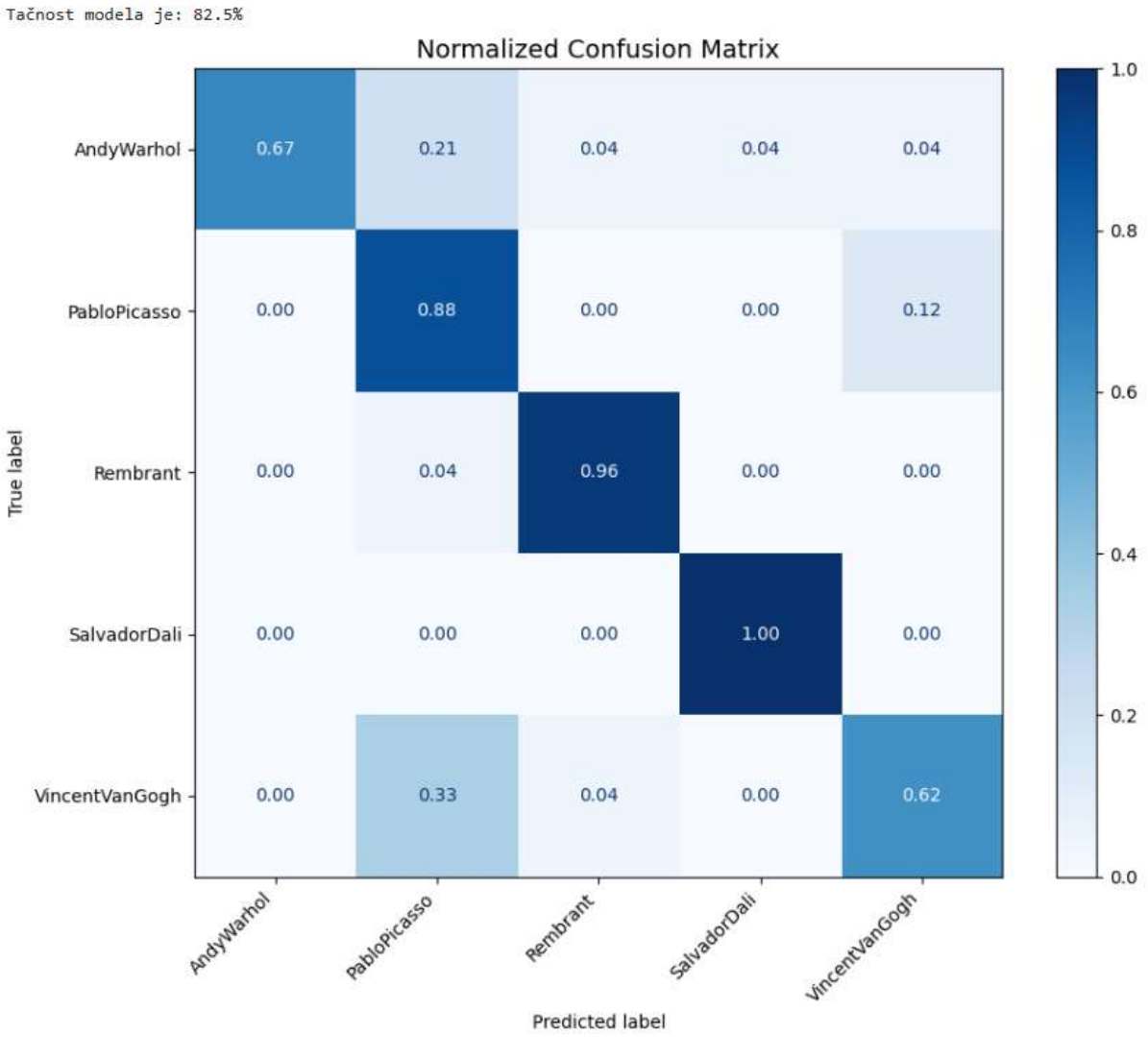


Slika18. Vrednosti performansi klasifikacije (tačnost, preciznost, osetljivost, f1-skor) za pojedinačne klase



Slika19. Tačnost i matrica konfuzije trening skupa

Slika20. Primer loše klasifikovane slike iz skupa podataka



Slika21. Tačnost i matrica konfuzije test skupa

Slika22. Primer dobro i loše klasifikove slike iz skupa podataka

Zadatak 3

Projektovanje fuzzy regulatora

Opredeliti se za jedan od pristupa projektovanju fuzzy upravljanja: intuitivni ili fazifikacija konvencionalnog upravljanja. U skladu sa opredeljenjem, projektovati po izboru jedan sistem fuzzy upravljanja za praćenje referentne vrednosti objekta upravljanja zadatog varijantom **S=5**. Postupak projektovanja, usvojenu strukturu i konkretno podešavanje parametara regulatora navesti u izveštaju. Napraviti Simulink model sistema upravljanja zadatim objektom u zatvorenoj sprezi. Realizovati odziv na step referentne vrednosti sa minimalne vrednosti na maksimalnu vrednost.

Opredeliti se za jedan od pristupa projektovanju fuzzy upravljanja: intuitivni ili fazifikacija konvencionalnog upravljanja, pa projektovati jedan sistem fuzzy upravljanja za potiskivanje nemerljivog poremećaja na ulazu objekta upravljanja zadatog varijantom S. Smatrati da je referentna vrednost uvek r = 0 i da je očekivana maksimalna amplituda poremećaja jednaka polovini maksimalne amplitude upravljačkog signala. Napraviti Simulink model sistema upravljanja zadatim objektom u zatvorenoj sprezi. Realizovati odziv na sledeći vremenski profil poremećaja: step sa nulte na maksimalnu moguću vrednost, pa, nakon smirivanja tranzijenta, step na minimalnu moguću vrednost.

Od rezultata prikazati i komentarisati vremenske oblike:

• signala upravljanja,

• regulisane varijable (signala na izlazu objekta upravljanja),

• signala na neposrednom ulazu u fuzzy inference sistema za oba sistema.

Na osnovu dobijenih rezultata sumirati osobine projektovanih sistema upravljanja.

Problem

5	$G(s) = \frac{0.07-0.1s}{(s+0.1)^2}e^{-3s}$	$r \in [-2, +2]$	$u \in [-1, +1]$

- **Referenca (r)** može ići od –2 do +2.
- **Upravljački signal (u)** mora ostati u granicama –1 do +1

Prvi deo

Za dati **objekat upravljanja** pravimo fuzzy regulator koji „tera“ izlaz sistema da prati zadatu vrednost **opseg referenci** (step od minimalne do maksimalne vrednosti), dok drži upravljački signal u zadatim granicama upravljačkog signala - **ograničenja upravljanja** i radi sa zadatim kašnjenjem.

Drugi deo

Za dati **objekat upravljanja,** sa stalnom referencom nula i očekivanom maksimalnom amplitudom poremećaja jednakoj polovini maksimalne amplitude upravljačkog signala, pravimo fuzzy regulator koji odbacuje nemerljiv poremećaj sa profilom „skok na maksimum“ pa „skok na minimum“. Cilj je brzo vraćanje izlaza na nulu uz poštovanje **ograničenja upravljanja** i zadatog kašnjenja.

Intuitivni pristup vs. fazifikacija konvencionalnog upravljanja

Kod intuitivnog pristupa polazimo od nule, a pravila se formulišu ručno, po osećaju. Jednostavan je i fleksibilan, ali se teško precizno podešava, posebno za složene sisteme.

Kod fazifikacije konvencionalnog upravljanja polazimo od poznate strukture tj. postojećeg konvencionalnog kontrolera (PID, PD,...), pa „fazifikujemo“ njegove komponente. Ponašanje mu je stabilnije, zahteva malo više teorije i podešavanja, ali daje mnogo bolje performanse.

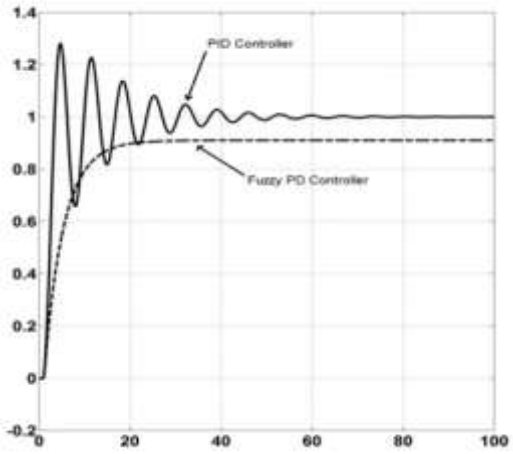
Zbog svega navedenog, opredelili smo se za fazifikaciju konvencionalnog upravljanja.

PID i fuzzy PD

Kladični PID ima tri dela:

- **P** (proporcionalni): reaguje na trenutnu grešku
- **I** (integralni): akumulira prošle greške (smanjuje stacionarnu grešku, ali zna da uspori sistem i izazove oscilacije)
- **D** (derivativni): reaguje na promenu greške (ublažava oscilacije i poboljšava brzinu odziva).

Kada uvedemo fuzzy komponentu, dovoljno je koristi **samo P i D deo**, **bez I komponente**, jer fuzzy sistem već “meko” reaguje.



Slika23. Poređenje klasičnog PID i fuzzy PD kontrolera

Dakle, koristimo fuzzy PD kontroler.

Fuzzy PD i fazifikaciju konvencionalnog upravljanja

Projektujemo fuzzy sistem tako da imitira ponašanje PD kontrolera, pri čemu su proporcionalna i derivativna akcija implementirane kroz fuzzy logiku, a ne klasičnu matematičku relaciju.

$$u(t) = K_p e + K_d \frac{de}{dt}$$

Slika23. Formula klasičnog PD kontrolera

Odnosno, umesto fiksnih K_p i K_d, pravimo fuzzy sistem koji dinamički menja vrednost u zavisnosti od situacije.

Fuzzy pravila

Kod npr. PID kontrolera se izlaz računa po formuli, dok fuzzy PD koristi **pravila.**

Npr. IF greška is velika pozitivna AND promena greške is mala negativna THEN izlaz is pozitivan veliki.

U(t)		Error change de/dt						
		NB	***NM***	***NS***	***ZO***	***PS***	***PM***	***PB***
error	***NB***	NB	NB	NB	NB	NB	NM	ZE
	NM	NB	NB	NB	NB	NM	ZE	PM
	NS	NB	NB	NB	NM	ZE	PM	PB
	ZE	NB	NB	NM	ZE	PM	PB	PB
	PS	NB	NM	ZE	PM	PB	PB	PB
	PM	NM	ZE	PM	PB	PB	PB	PB
	PB	ZE	PM	PB	PB	PB	PB	PB

Slika24. Tabela fuzzy pravila

Signal greške error je razlika između željene i stvarne vrednosti. Signal error change de/dt je brzina promene greške, a signal u izlaz (upravljanje).

Error e(t)		change in error de/dt		Controller output U(t)	
NB	Negative Big	***NB***	Negative Big	***NB***	Negative Big
NM	Negative Medium	***NM***	Negative Medium	***NM***	Negative Medium
NS	Negative Small	***NS***	Negative Small	***NS***	Negative Small
ZE	Zero	***ZE***	Zero	***ZE***	Zero
PS	Positive Small	***PS***	Positive Small	***PS***	Positive Small
PM	Positive Medium	***PM***	Positive Medium	***PM***	Positive Medium
PB	Positive Big	***PB***	Positive Big	***PB***	Positive Big

Slika25. Tabela lingvističkih vrednosti

Struktura sistema

Tip sistema: `Mamdani (type-1)`

Broj ulaza: `2 (error, de/dt)`

Broj izlaza: `1 (u)`

Funkcije pripadnosti: `triangular`

Broj lingvističkih vrednosti: `7 (NB, NM, NS, ZE, PS, PM, PB)`

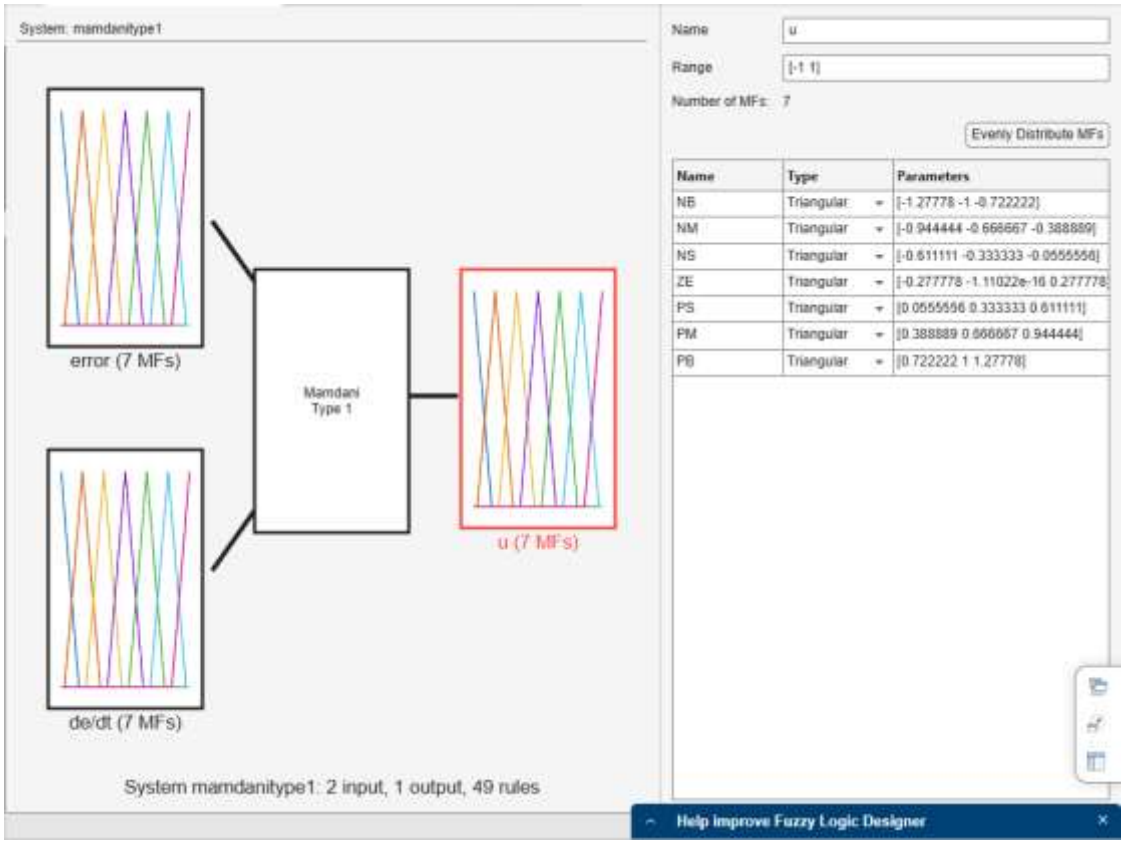
Broj pravila: `49`

Implikacija: `min`

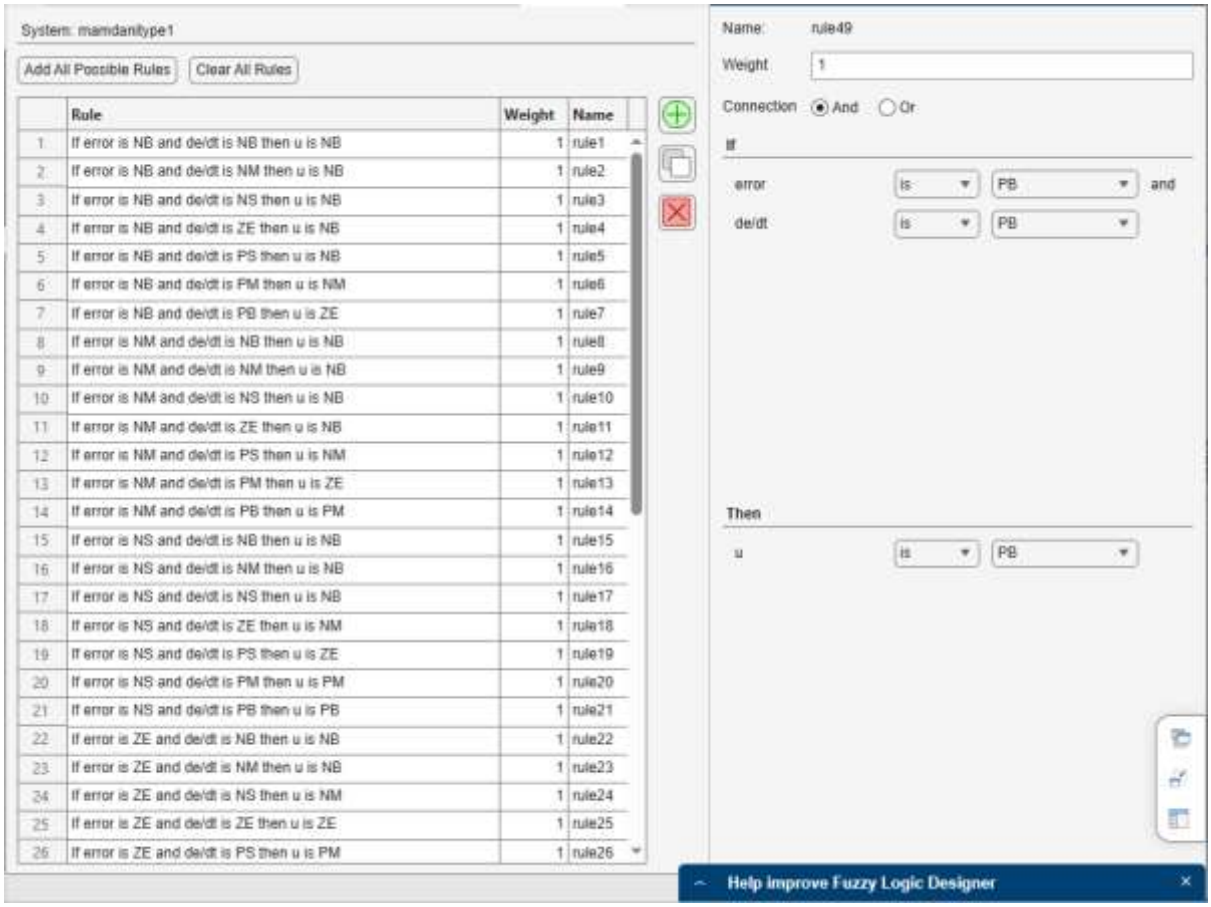
Agregacija: `max`

Defazifikacija: `centroid`

Greška error i de/dt su u realnom opsegu [-2,2], ali u fuzzy opsegu [-1,1], a izlaz tj. upravljanje u je i u realnom i u fuzzy opsegu [-1,1]. Ovaj deo ćemo kasnije regulisati gainovima pre FIS (Fuzzy Interface System) bloka.



Slika26. Podešavanje ulaza i izlaza u Matlab Online



Slika27. Podešavanje pravila u Matlab Online

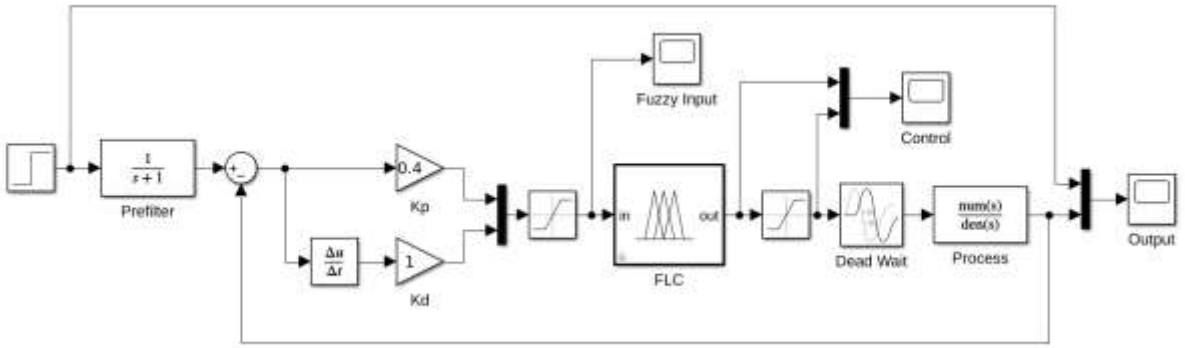
Simulink - Prvi deo

Cilj je napraviti **fuzzy PD regulator** koji:

- tera izlaz sistema y(t) da prati referencu r(t), gde je opseg reference [-2, +2],
- drži upravljački signal u(t) u granicama [–1, +1],
- uzima u obzir kašnjenje iz modela e^{-3s},
- radi za dati objekat upravljanja:

$$G(s) = \frac{0.07 - 0.1s}{(s + 0.1)^2} e^{-3s}$$

Šema



Slika28. Šema rešenja prvog dela zadatka

Za ovu šemu fuzzy PD regulatora postavili smo sledeće:

Process - Numerator coefficients: [-0.1 0.07], Denominator coefficients: [1 0.2 0.01]

Dead Wait - Time delay: 3

FLC - FIS name: 'fuzzyPDmodel.fis'

Kd i **Kp** - Pojačanja proračunavamo po pravilima. Najpre podešavanjem PD regulatora dobijamo stabilno ponašanje za:

$$K_p = 0.2, \quad T_d = 2.5$$

Dalje, kako je izlazni univerzum [-2, 2], a upravljanje ograničeno na opseg [-1, 1], uzima se:

$$\hat{K}_u = \frac{U_{\max}}{2} = \frac{1}{2} = 0.5$$

I konačno, ubacivanjem u formule:

$$\hat{K}_p = \frac{K_p}{\hat{K}_u},$$
$$\hat{K}_d = \frac{K_p T_d}{\hat{K}_u}$$

dobijaju se vrednosti:

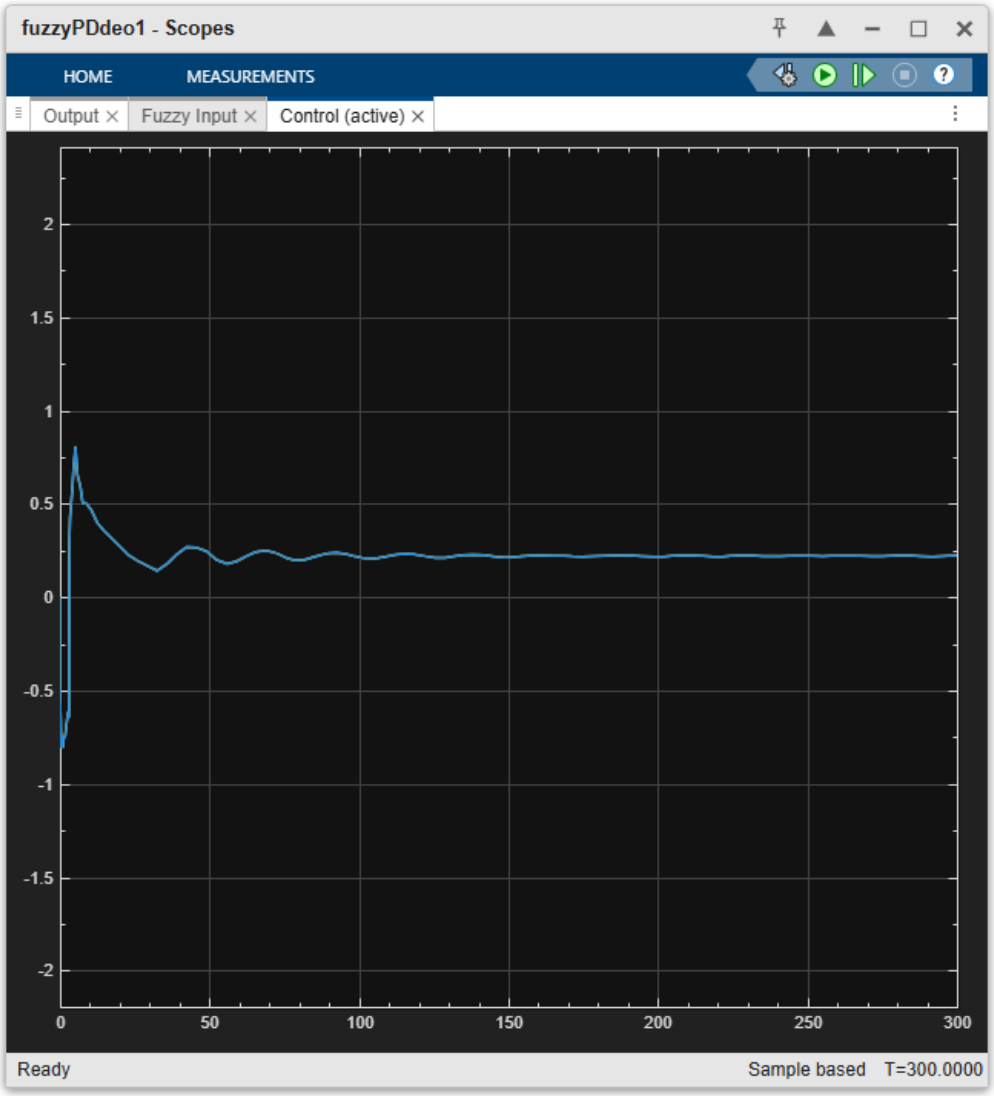
$$\boxed{\hat{K}_p = 0.4, \quad \hat{K}_d = 1}$$

Step - Initial value: -2.0, Final value: 2.0

Saturation - Upper limit: 1.0, Lower limit: -1.0

Prefilter - Koristimo standardni prefilter koji obrađuje referencu r(t) pre nego što uđe u regulator. Umesto da regulator vidi naglu skokovitu referencu (step), prefilter mu prosleđuje ublaženu „omekšanu“ verziju reference. Praktikuje se kod sistema sa većim transportnim kašnjenjem (u konkretnom slučaju transportno kašnjenje je 3 sekunde).

Vremenski oblici signala

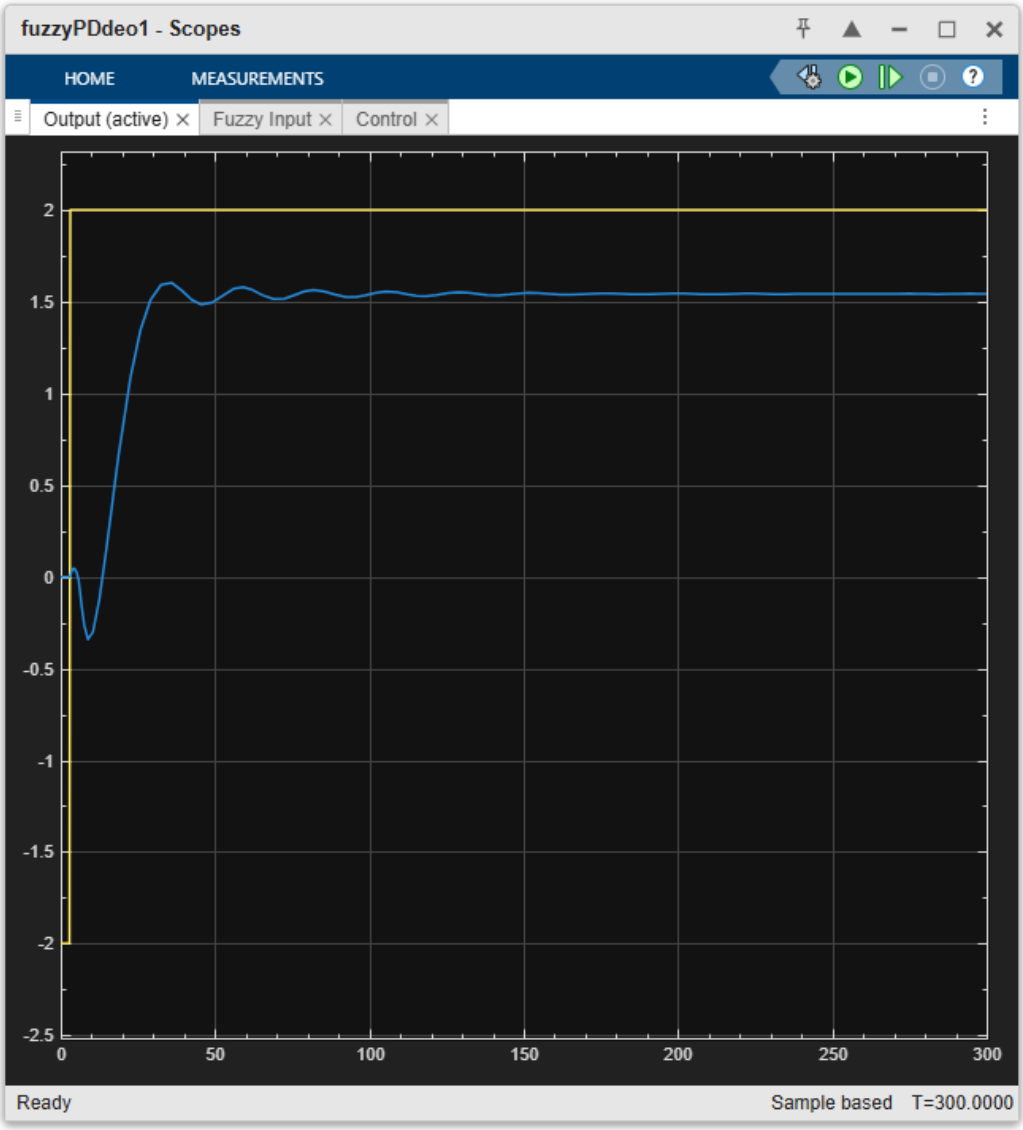


Slika29. Signal upravljanja (**Control**)

Možemo primetiti da se dva prikazana vremenska oblika na ovom grafiku (pre i posle Saturation bloka) u potpunosti preklapaju, dakle, ovaj Saturation blok u posmetranom slučaju

nema nikakav efekat, ali je svakako tu radi prevencije dolaska vrednosti van očekivanog opsega na ulaz narednih komponenti u šemi.

Na početku simulacije upravljački signal naglo raste usled skokovite promene reference (kašnjenje je 3 sekunde). Vrednost signala brzo dostiže granične vrednosti opsega (-1 i 1), što ukazuje da fuzzy PD koristi maksimalno dostupan upravljajući napor kako bi kompenzovao veliko kašnjenje procesa. Nakon toga, signal se stabilizuje i ostaje gladak, bez izraženih oscilacija, što pokazuje da je upravljanje umereno i da ne ulazi u zasićenje. Prefilter koji smo dodali je ublažio inicijalni skok reference, pa je signal upravljanja kontinuiran i bez oštrih promena.

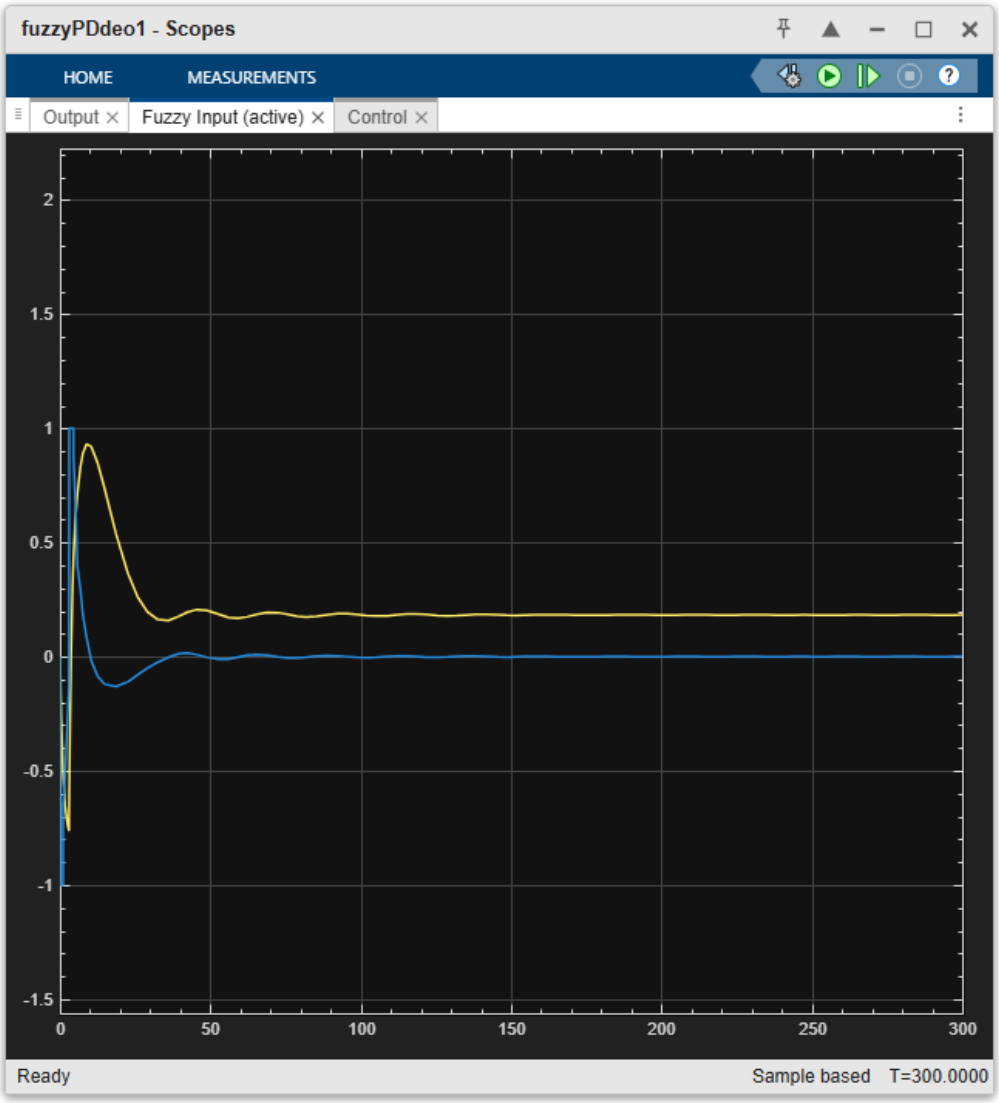


Slika30. Regulisana varijabla (**Output**)

Izlaz procesa (označen **plavom** bojom) prati zadatu **referencu** (označenu **žutom** bojom) uz mali preskok i brzo smirenje. Zbog transportnog kašnjenja (3 sekunde) sistem ne može

trenutno da reaguje, ali zahvaljujući diferencijalnom dejstvu kontrolera oscilacije se brzo prigušuju.

Regulisana varijabla se stabilizuje oko zadate vrednosti bez trajnog odstupanja, što pokazuje da je fuzzy kontroler efikasno kompenzovao kašnjenje i nelinearnosti procesa.



Slika31. Signal na neposrednom ulazu u fuzzy inference (**Fuzzy Input**)

Na grafiku sa slike prikazani su vremenski oblik signala:

Kp · e (označen **žutom** bojom) i **Kd · (de/dt)** (označen **plavom** bojom).

Greška e(t) ima najveću vrednost pri početnom skoku i postepeno opada ka nuli kako sistem prati referencu. Promena greške de/dt ima karakter oštrijeg impulsa na početku, zatim osciluje i brzo se smiruje.

Sa grafika možemo zaključiti da su ulazne promenljive fuzzy kontrolera dobro skalirane i da sistem radi u okviru definisanih univerzuma [−1, 1].

Zaključak

Projektovani fuzzy PD kontroler obezbeđuje:

- **brz i gladak odziv** bez značajnog preskoka,
- **stabilno ponašanje** uprkos velikom kašnjenju procesa,
- **ograničen i kontinuiran signal upravljanja** u opsegu [−1, 1],
- **rad ulaznih signala fuzzy inference sistema u celom efektivnom opsegu** [−1, 1].

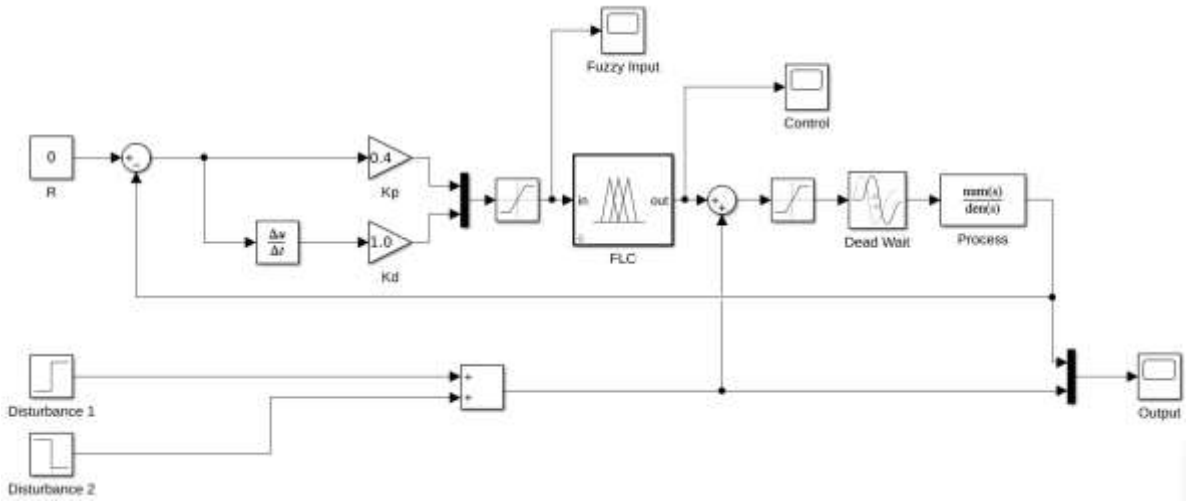
Da sumiramo, sistem ima dobru stabilnost, umerenu brzinu odziva i dobro toleriše kašnjenje, što potvrđuje da je postignut cilj fuzzy upravljanja – poboljšane su performanse u odnosu na klasični PD regulator.

Simulink - Drugi deo

Cilj je dodati poremećaj na ulaz objekta upravljanja i projektovati fuzzy PD regulator koji će efikasno potisnuti taj poremećaj, tako da izlaz sistema ostane što bliži referenci (r = 0), uz poštovanje ograničenja upravljačkog signala i transportnog kašnjenja.

Drugim rečima, zadatak ovog modela je da se ispita sposobnost fuzzy PD regulatora da brzo stabilizuje signal nakon pojave poremećaja tipa „skok na maksimum – skok na minimum“.

Šema



Slika28. Šema rešenja drugog dela zadatka

Ovde ne koristimo prefilter za „omekšavanje“ reference, jer je ona konstantna i jednaka nuli. Za razliku od prethodne šeme, postavili smo i sledeće:

Constant Referenca (**R**) - stalna je i jednaka je nuli, Constant value = 0.

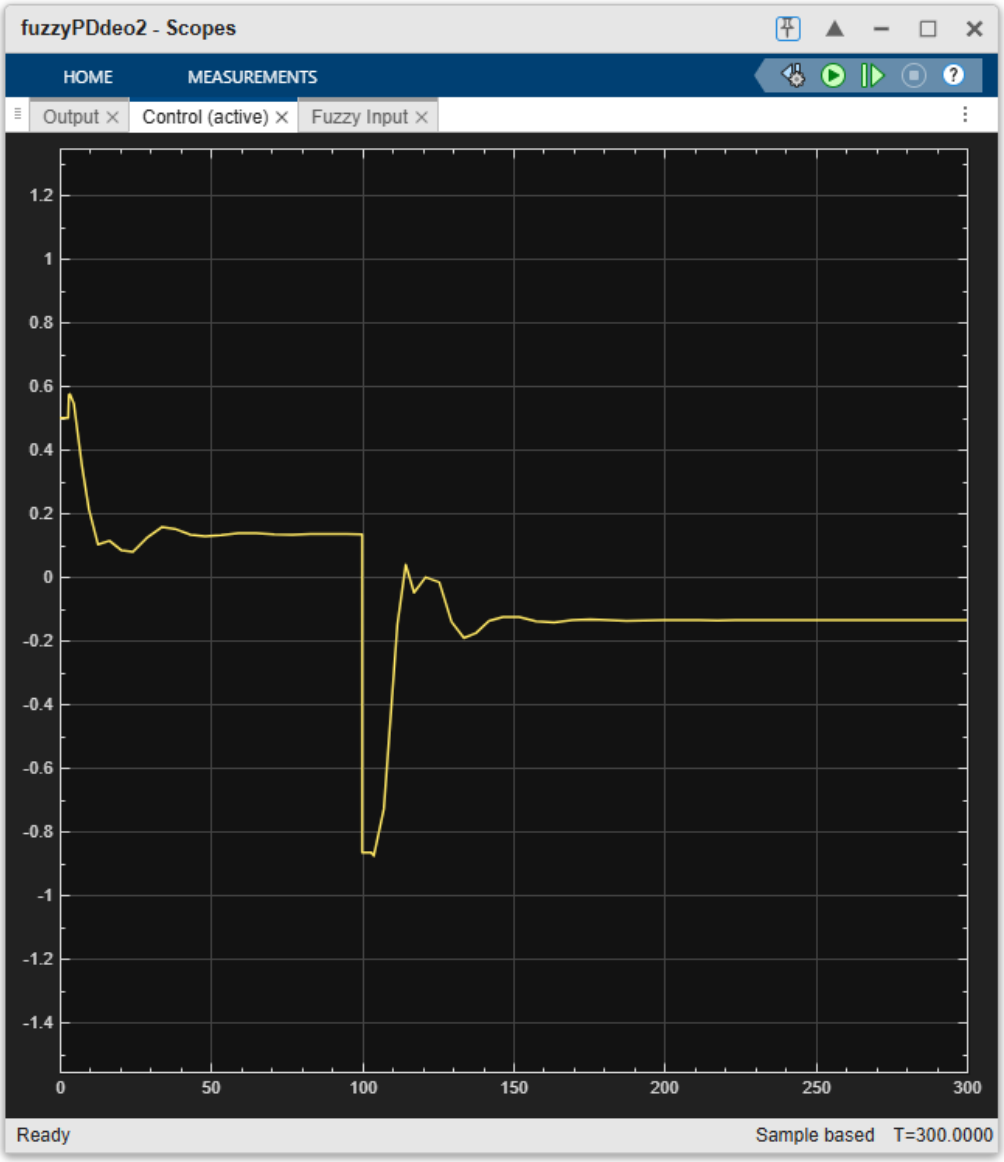
Prvi **Step** blok (**Disturbance 1**) - generiše skok sa nule na maksimalnu vrednost poremećaja. Prema tekstu zadatka maksimalna amplituda poremećaja jednaka je polovini maksimalne amplitude upravljačkog signala, odnosno:

Step time = 0, Initial value = 0, Final value = +0.5.

Drugi **Step** blok (**Disturbance 2**) - aktivira se nakon što se sistem stabilizuje posle prvog skoka. Generiše skok sa nule na minimalnu vrednost poremećaja. Da bi se skočilo na minimalnu vrednost, potrebno je najpre spustiti se na nulu, a zatim na vrednost -0.5, pa je:

Step time = 100, Initial value = 0, Final value = -1.0.

Vremenski oblici signala

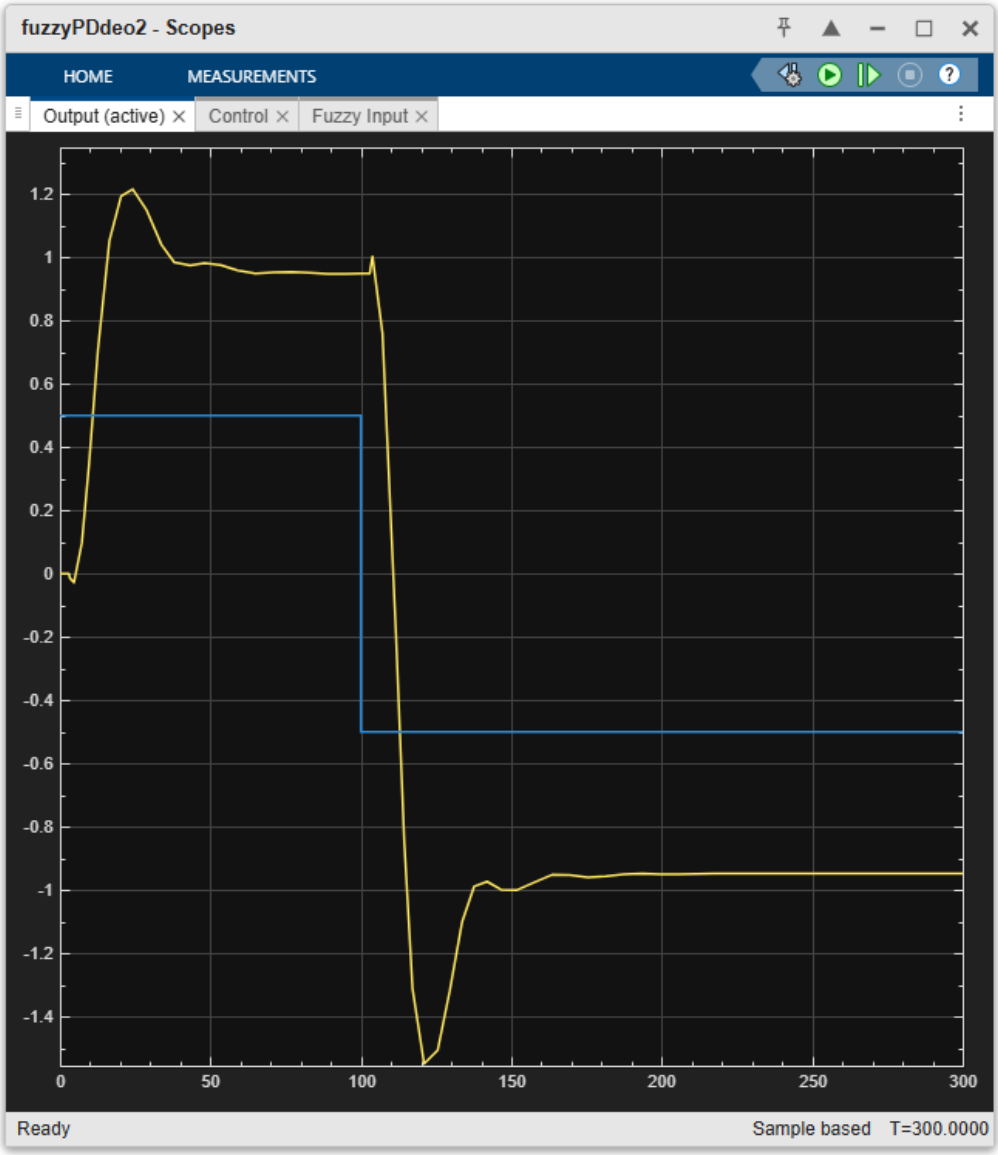


Slika29. Signal upravljanja (**Control**)

Na početku svakog poremećaja (trenuci 0 i 100) uočava se nagla promena upravljačkog signala, što je očekivano ponašanje fuzzy PD kontrolera - on momentalno reaguje na naglu promenu greške.

Upravljanje kratkotrajno dostiže granice dozvoljenog opsega, ali se zatim brzo vraća u centralnu zonu. Nakon stabilizacije sistema, signal osciluje oko nule sa malom amplitudom, što ukazuje na dobro prigušavanje i odsustvo trajne greške.

Dakle, odziv je brz, bez značajnog prekoračenja, a regulator koristi ceo opseg upravljanja.



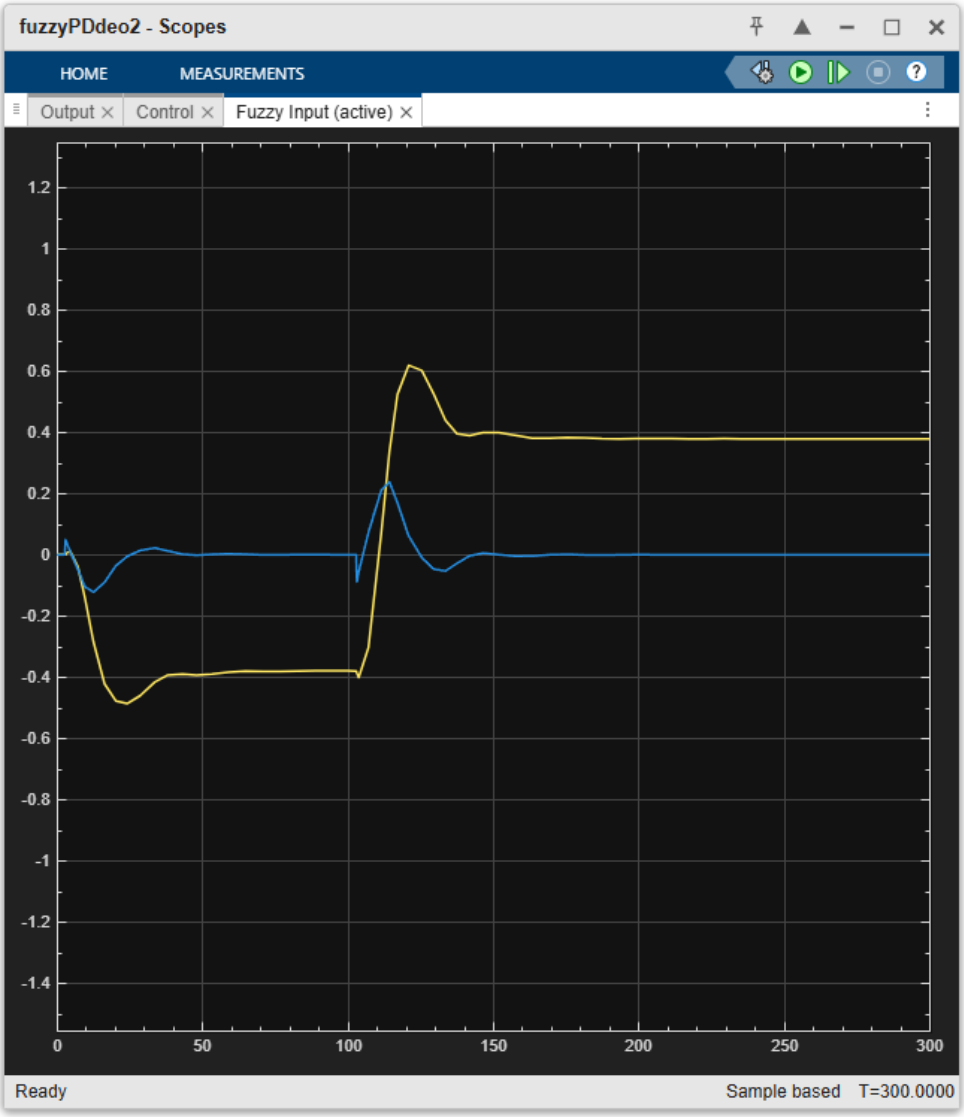
Slika30. Regulisana varijabla (**Output**)

Na grafiku sa slike prikazani su vremenski oblik signala:

Output (označen **žutom** bojom) i **suma Step elemenata** (označena **plavom** bojom).

Izlaz sistema posle prvog pozitivnog poremećaja pokazuje nagli skok, nakon čega se u periodu od nekoliko sekundi vrednost stabilizuje.
Kod drugog, negativnog poremećaja, odziv je gotovo simetričan - izlaz odstupi u suprotnom smeru, ali se opet brzo stabilizuje.
U oba slučaja vreme smirenja je kratko, a preskok mali.

Možemo zaključiti da fuzzy PD regulator obezbeđuje efikasno potiskivanje poremećaja, brz odziv i stabilan sistem sa minimalnim oscilacilovanjem u stacionarnom stanju.



Slika31. Signal na neposrednom ulazu u fuzzy inference (**Fuzzy Input**)

Na grafiku sa slike prikazani su vremenski oblik signala:

Kp · e (označen **žutom** bojom) i **Kd · (de/dt)** (označen **plavom** bojom).

Signal greške e(t) prati oblik poremećaja, ali sa značajno manjom amplitudom zahvaljujući delovanju fuzzy kontrolera.
Pri pojavi skoka poremećaja, greška naglo poraste (ili opadne), a izvod greške de/dt dobija izražen impuls koji aktivira odgovarajuća fuzzy pravila.
Nakon što se sistem smiri, oba signala se stabilizuju.

Dakle, fuzzy PD regulator efikasno pretvara trenutne promene greške i brzinu promene greške u korektivno upravljanje.

Zaključak

Na osnovu simulacionih rezultata mogu se izdvojiti sledeće karakteristike fuzzy PD kontrolera:

- **Brza reakcija** na promenu poremećaja i **kratko vreme smirenja**.
- **Simetričan odziv** pri pozitivnom i negativnom poremećaju.
- **Bez značajnog preskoka** ni pri promeni znaka poremećaja.
- **Stabilnost** sistema očuvana i uz prisustvo transportnog kašnjenja.
- **Nema trajne greške** u stacionarnom režimu.