Konvoliuciniai neuroniniai tinklai

Pijus Petkevičius

2023 m. lapkričio 10 d.

Turinys

1 Įvadas

1.1 Tikslas

Užduoties tikslas - apmokyti konvoliucinį neuroninį tinklą KMNIST duomenimis ir palyginti skirtingų architektūrų, išmetimo sluoksnių skaičiaus, aktyvacijos funcijos, optimizavimo algoritmo, duomenu normalizavimo įtaką klasifikavimo tikslumui ir paklaidai.

1.2 Uždaviniai

- Paruošti duomenis tinklo apmokymui: duomenis perskirstyti į mokymo ir testavimo aibes, normalizuoti.
- Sukurti programinį kodą, kuriame būtų galima apmokyti tinklą KMNIST duomenimis ir ir testuoti apmokyto tinklo klasifikavimą.
- Atlikti tyrimus su konvoliuciniais neuroniniais tinklais:
  - Palyginti trijų skirtingų architektūrų klasifikavimo tikslumo rezultatus ir paklaidą.
  - Palyginti architektūros klasifikavimo tikslumo ir paklaidos rezultatus, kai nepridedamas išmetimo sluoksnis, pridedamas vienas išmetimo sluoksnis ir kai pridedami du išmetimo sluoksniai, taip pat, kai išmetimo sluoksnio tikimybės skirtingos.
  - Palyginti architektūros klasifikavimo tikslumo ir paklaidos rezultatus, kai duomenys yra normalizuojami.
  - Palyginti architektūros klasifikavimo tikslumo ir paklaidos rezultatus, kai naudojami skirtingos aktyvacijos funkcijos.
  - Palyginti architektūros klasifikavimo tikslumo ir paklaidos rezultatus, kai naudojami skirtingo optimizavimo algoritmai.

2 Eksperimentų vykdymas

2.1 Duomenys

Šiam darbui naudoti KMNIST Dataset rinkinio duomenys. Šiame rinkinyje yra 70000 paveikslėlių, iš viso yra 10 klasių, po 7000 paveikslėlių kiekvienoje klasėje. Kiekvienas paveikslėlis yra 28x28 pikselių dydžio.

Duomenų šaltinyje duomenys pateikti keturiuose failuose, du failai mokymo ir testavimo duomenų aibės, atitinkamai po 60000 ir 10000 duomenų. Kituose dviejuose failuose yra mokymo ir testavimo duomenų aibės klasių reikšmės.

2.1.1 Duomenų paruošimas

Duomenų paruošimui buvo sukurta keletas funkcijų, kurios paimdavo duomenų aibių dalis ir paruošdavo į 70000x28x28x1 masyvus. Vėliau, sujungus visas duomenų aibes, duomenys buvo padalinti į mokymo, validavimo ir testavimo aibes santykiu 80:10:10.

Duomenų klasių aibei panaudota funkcija `reshapeLabels(labels:list, numClasses:int)`, kuri duotas klasių reikšmes $(0; 9)$ paverčia į 10 ilgio binarinius vektorius, kur vienetukas yra tame indekse, kokia duomenų įeities klasė.

Kadangi pikselių reikšmės varijuoja intervale $[0; 255]$, duomenų įeitys buvo normalizuotos naudojantis formule $(x \div 255)$.

2.2 Parametrai

2.2.1 Konvoliucinio neuroninio tinklo architektūra

LeNet architektūros implementacija buvo nukopijuota iš BIGBALLON / cifar-10-cnn. Šią architektūrą sudaro:

- 2 konvoliuciniai sluoksniai;
- 3 pilnai sujungti sluoksniai.

Simple architektūros implementacija buvo nukopijuota iš tensorflow / docs. Pakeistas optimizavimo algoritmas iš Adam į SGD, kad būtų suvienodinti hiperparametrai su LeNet naudojamais hiperparametrais. Simple architektūrą sudaro:

- 3 konvoliuciniai sluoksniai;
- 2 pilnai sujungti sluoksniai.

AlexNet architektūros implementacija buvo nukopijuota iš the clever programmer. AlexNet architektūrą sudaro:

- 5 konvoliuciniai sluoksniai;
- 2 pilnai sujungti sluoksniai.

2.2.2 Hiperparametrai

- epochų skaičius - 15;
- mokymo greitis, $\eta = 0{,}005$;
- paketo dydis - 100;
- išmetimo sluoksnio tikimybė - 0,2;
- optimizavimo algoritmas - stochastinis gradientinis nusileidimas (SGD) su momentum 0,9;
- nuostolių funkcija - *categorical crossentropy*.

2.3 Skaičiavimo resursai

Naudoti debesijos sprendimai „Google Colab“ aplinkoje, GPU versija.

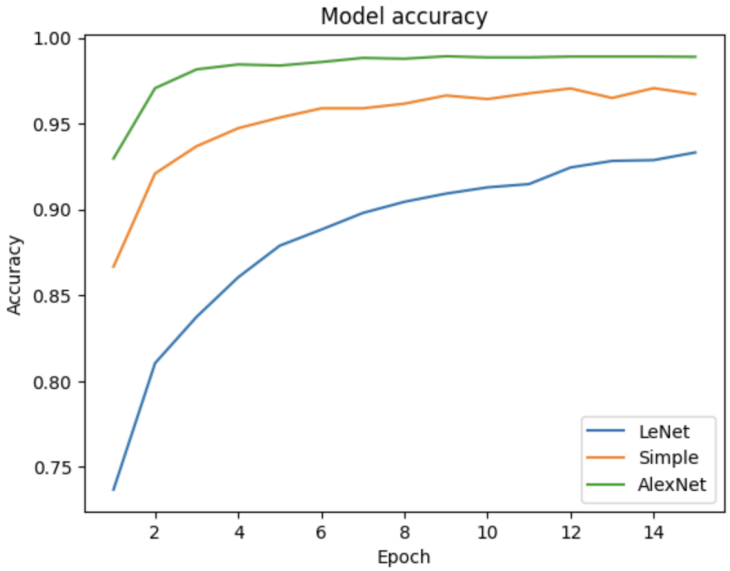
3 Programos kodas

Programos kodas gali būti rastas šioje „Google Colab“ aplinkoje.

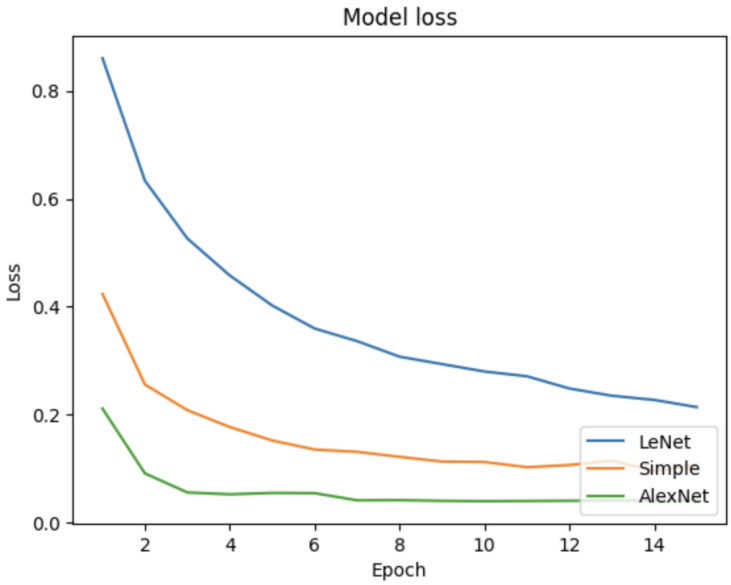
4 Rezultatai

4.1 Skirtingų architektūrų palyginimas

Šio tyrimo metu palygintos trys architektūros: LeNet, Simple ir AlexNet. diagramose galime matyti, kad AlexNet stipriai pirmauja tikslumo ir paklaidos grafikuose (1 pav. tikslumas >90 %). Nuo 9 epochos AlexNet pasiekia 100% tikslumą, kol Simple ir Lenet yra pasiekę 88-95% tikslumą.



1 pav.: Skirtingų architektūrų klasifikavimo tikslumas



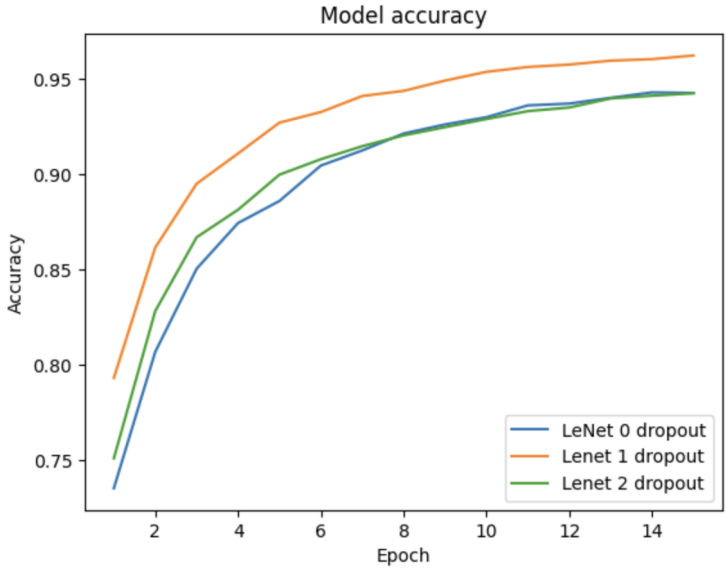
2 pav.: Skirtingų architektūrų klasifikavimo paklaida

4.2 Skirtingų išmetimo sluoksnių skaičiaus ir išmetimo sluoksnio tikimybių palyginimas

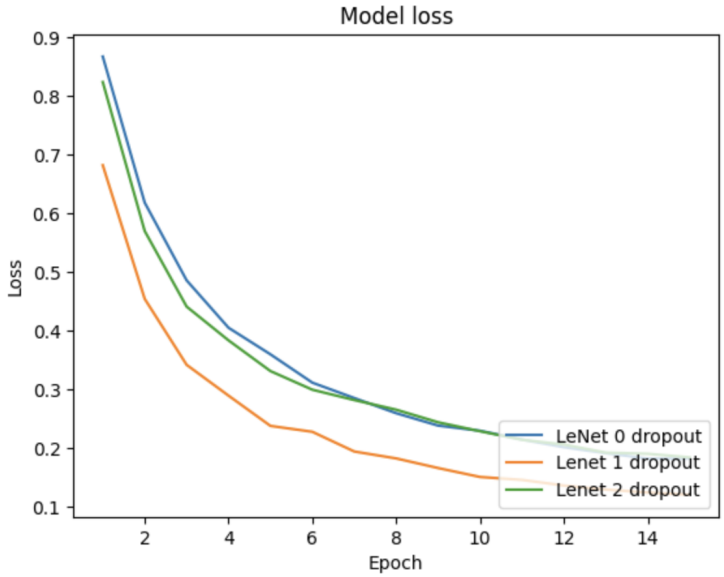
Šio tyrimo metu atlikti bandymai su LeNet architektūra(prasčiausiai pasirodžiusia iš turimų trijų) ir ji papildyta 1 arba 2 išmetimo sluoksniais. Vieną išmetimo sluoksnį turinti architektūra išmetimo sluoksnį turėjo po pirmojo konvoliucinio sluoksnio, du išmetimo sluoksnius turinti architektūra išmetimo sluoksnius turėjo ir po pirmojo konvoliucinio sluoksnio, ir po pirmojo pilnai sujungto sluoksnio. Taip pat buvo atliekamas tyrimas su skirtingomis išmetimo sluoksnio tikimybių reikšmėmis $\{0{,}2; 0{,}4; 0{,}6\}$.

LeNet tinklo didžiausias tikslumas buvo pasiektas turint 1 išmetimo sluoksnį. Tai galime matyti 3 pav.

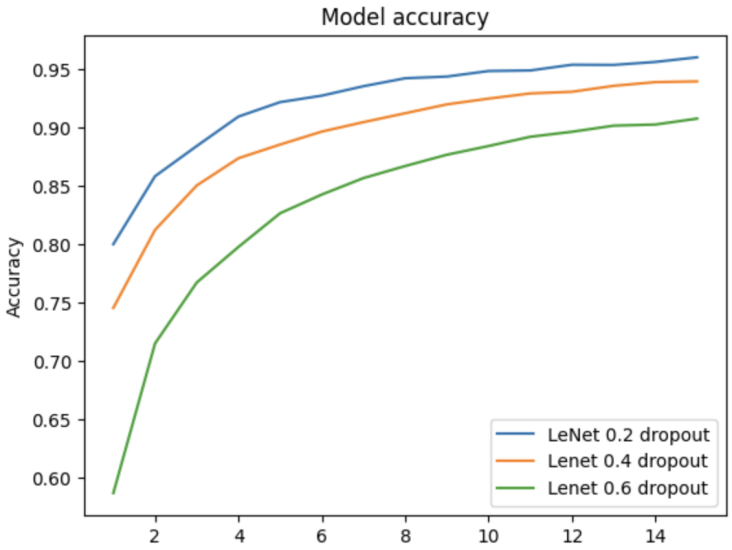
5 pav. buvo lyginamos skirtingos išmetimo sluoksnio tikimybės reikšmės. Buvo naudojana LeNet achitektūra su 2 išmetimo sluoksniais. Matome, kad esant mažesnei išmetimo sluoksnio tikimybei, klasifikavimo tikslumas yra didžiausias.



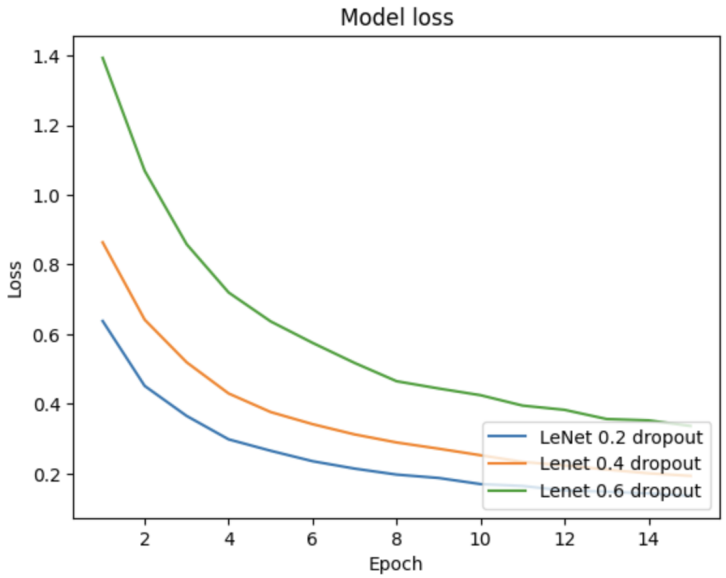
3 pav.: LeNet architektūros su skirtingu išmetimo sluoksnių skaičiumi klasifikavimo tikslumas



4 pav.: LeNet architektūros su skirtingu išmetimo sluoksnių skaičiumi klasifikavimo paklaida



5 pav.: LeNet architektūros su skirtingu išmetimo sluoksnio tikimybėsmis klasifikavimo tikslumas

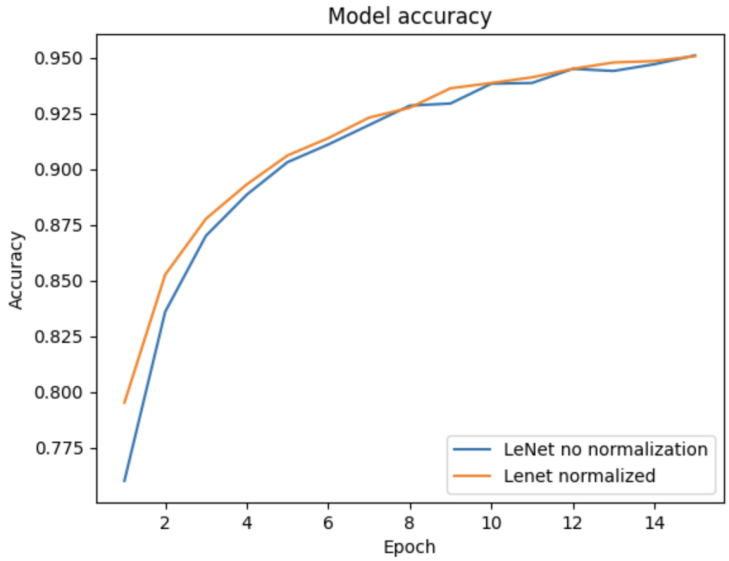


6 pav.: LeNet architektūros su skirtingu išmetimo sluoksnio tikimybėmis klasifikavimo paklaida

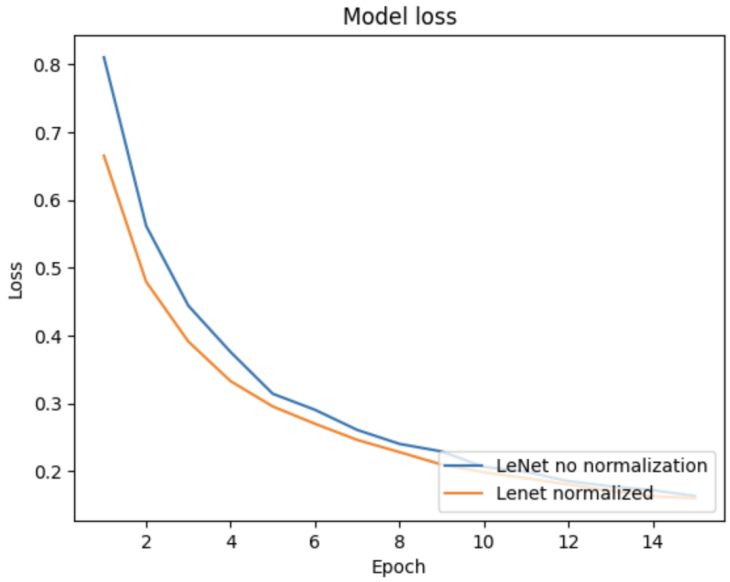
4.3 Duomenų normalizavimo įtaka duomenų klasifikavimo tikslumui ir paklaidai

Šio tyrimo metu atlikti bandymai su LeNet architektūra ir ji papildyta normalizacijos sluoksniais po kiekvieno konvoliucinio sluoksnio.

7 pav. galime matyti, kad esant normalizacijos sluoksniui po kiekvieno konvoliucinio sluoksnio pradinis tikslumas yra aukštesnis(76%;79,51%), paklaida mažesnė (0,6654). 15 epochos metu, normalizuotos ir nenormalizuotos LeNet architektūros pasiekia tą patį tikslumą ir paklaidą.



7 pav.: LeNet architektūros su ir be normalizacijos sluoksniu klasifikavimo tikslumas

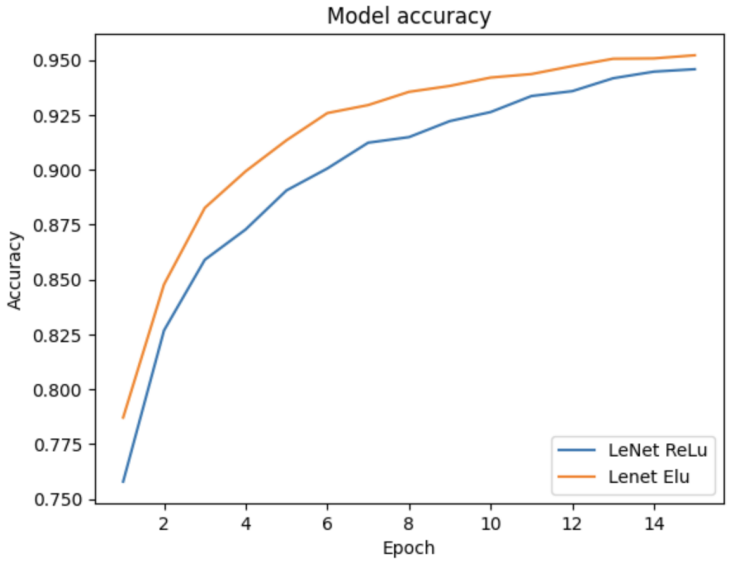


8 pav.: LeNet architektūros su ir be normalizacijos sluoksniu klasifikavimo paklaida

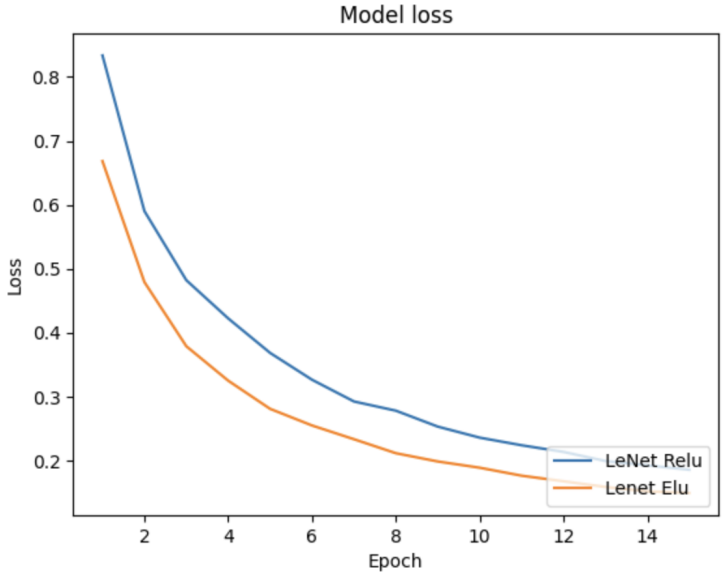
4.4 Aktyvacijos funkcijos įtaka duomenų klasifikavimo tikslumui ir paklaidai

Šio tyrimo metu atlikti bandymai su LeNet architektūra ir su skirtingomis aktyvacijos funkcijomis: Elu ir ReLu.

9 pav. galime matyti, kad naudojant Elu funkciją modelio pradinis tikslumas yra aukštesnis nei Relu funkcijos(76%; 79%) ir išsilaiko iki 15 epochos (93%; 95%).



9 pav.: LeNet architektūros su ReLu ir Elu aktyvacijos funkcija klasifikavimo tikslumas

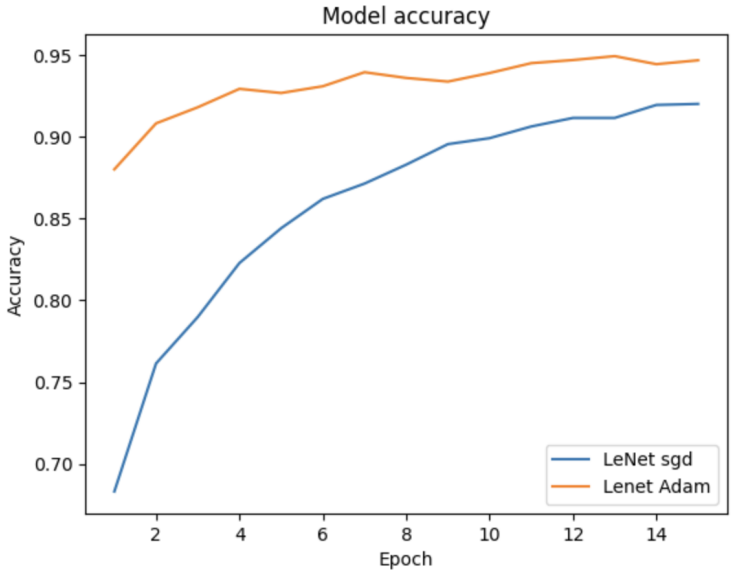


10 pav.: LeNet architektūros su ReLu ir Elu aktyvacijos funkcija klasifikavimo paklaida

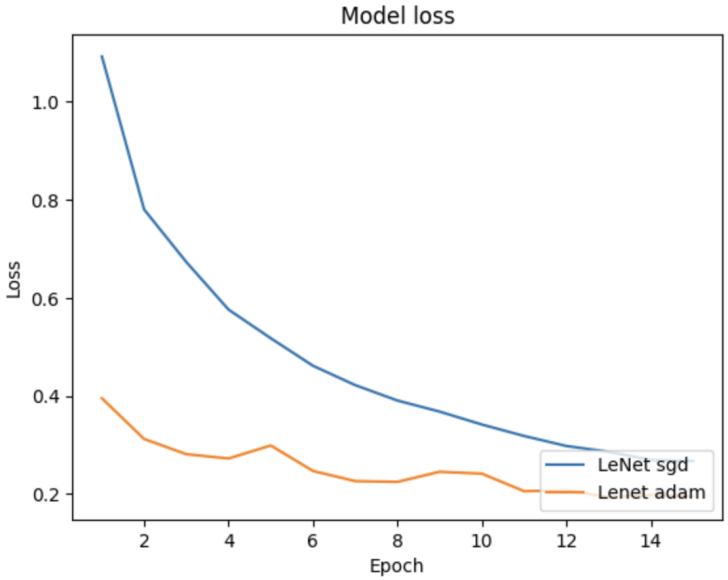
4.5 Optimizavimo algoritmo įtaka duomenų klasifikavimo tikslumui ir paklaidai

Šio tyrimo metu atlikti bandymai su LeNet architektūra ir su skirtingais optimizavimo algoritmais: sgd ir adam.

11 pav. galime matyti, kad naudojant Adam optimizavimo algoritmą, modelio pradinis tikslumas yra žymiai aukštesnis nei Sgd optimizavimo algoritmo (68,32%; 88,02%) ir po 15 epochų Adam tikslumas išlieka didesnis (92,02%; 94,68%).



11 pav.: LeNet architektūros su sgd ir adam optimizavimo algoritmais klasifikavimo tikslumas

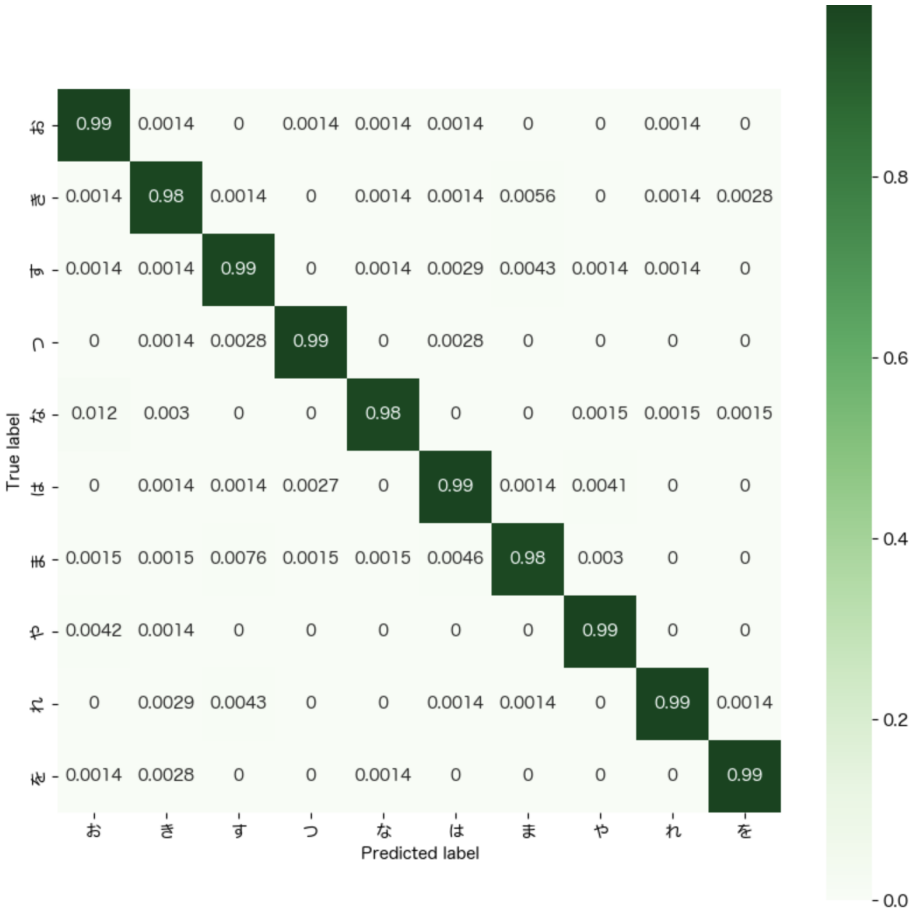


12 pav.: LeNet architektūros su sgd ir adam optimizavimo algoritmais klasifikavimo paklaida

4.6 Geriausias rezultatas

Geriausias rezultatas pagal klasifikavimo tikslumo kriterijus pasiektas su AlexNet architektūra ir 9 epochų. Šio bandymo validavimo duomenų klasifikavimo tikslumas yra 0,9866, paklaida 0,0421.

Su šiuo apmokytu modeliu testavimo duomenims sudaryta klasifikavimo matrica. 13 pav. matyti, kad prasčiausiai suklasifikuota ま, き, す simboliai. ま klasei priklausančių 0,0076 duomenų buvo priskirta す , き klasei priklausančių 0,0056 duomenų buvo priskirta ま , o す klasei priklausančių 0,0043 duomenų buvo priskirta ま .



13 pav.: AlexNet architektūros testavimo duomenų klasifikavimo matrica

Taip pat iš testavimo duomenų aibės parinkta 30 įrašų, kur būtų bent vienas įrašas iš kiekvienos klasės. Šiems įrašams nurodyta tikroji ir nustatyta klasė 1 lentelėje. Įrašai, kuriems nustatyta klaidinga klasė, paryškinti. Iš duotų 30 įrašų trims nustatyta klaidinga klasė, vadinasi, klasifikavimo tikslumas šiems duomenis yra apie 0,90.

1 lentelė: Testavimo duomenų tikrosios klasės ir AlexNet nustatytos klasės

Nr.	**Tikroji klasė**	**Nustatyta klasė**
1	お	お
2	つ	つ
3	を	を
4	な	な
5	は	は
6	お	お
7	**は**	**ま**
8	お	お
9	は	は
10	つ	つ
11	を	を
12	お	お
13	な	な
14	つ	つ
15	す	す
16	を	を
17	つ	つ
18	つ	つ
19	を	を
20	**ま**	**は**
21	き	き
22	**す**	**れ**
23	れ	れ
24	つ	つ
25	は	は
26	つ	つ
27	き	き
28	き	き
29	す	す
30	れ	れ

.

14 pav.: Testavimo duomenų tikrosios klasės ir AlexNet nustatytos klasės

5 Išvados

- Išmetimo sluoksniai LeNet architektūros pasiektą tikslumą suprastino, tačiau ties 24-30 epochomis padėjo LeNet architektūrai nepersimokyti ir išlaikyti nekylančią paklaidą. Tuo tarpu atliekant eksperimentą su LeNet architektūra be išmetimo sluoksnių paklaida ėmė kilti ties 24 epocha.
- AlexNet architektūra turėjo žymiai aukštesnį tikslumo lygį pirmosiomis epochomis ir pakanka 9 epochų pasiekti 100% tikslumą.
- 1 išmetimo sluoksnis padėjo padidinti LeNet tinklo tikslumą, tuo tarpu išmetimo sluoksnio tikimybei augant konvoliucinio neuroninio tinklo tikslumas mažėjo.
- Duomenų normalizavimas gana didelę įtaką turi pirmomis 5 epochomis, vėliau nenormalizuotų duomenų tikslumas tampa gana panašus su normalizuotais.
- Elu aktyvacijos funkcija turėjo žymiai geresnį tikslumą, lyginant su ReLu aktyvacijos funkcija.
- Adam optimizavimo algoritmais stipriai lenkė sgd optimizavimo algoritmą, pirmosiomis epochomis skirtumas buvo iki 10% tikslumo.
- Geriausiai pasirodęs modelis atsižvelgiant į klasifikavimo tikslumą buvo AlexNet architektūros modelis, mokytas 9 epochas. Jo klasifikavimo tikslumas yra 0,9866, o paklaida - 0.0421.
- Atrinktam AlexNet architektūros modeliui sunkiausiai sekėsi atskirti ま, き, す simbolius.