Detectia Tumorilor utilizand Retele Neurale

Student Pietraru Alfred Andrei
Grupa 343C2

1 Cerinta 1

Aici am implementat pipeline-ul care se va executa pentru cross validation. Si am urmatoarele functii: split_for_cross_validation, care primeste totalitatea datelor de antrenament si le imparte pe acestea in k-fold-uri cu numar egal de elemente si pastrand distributia claselor in fiecare fold. Iar cealalta functie importanta este compute_cross_validation, care primeste mai multi parametrii printre care: parametrii modelului pe care il antreneaza, K care este numarul de fold-uri, cele K fold-uri calculate de catre functia split_for_cross_validation, operatiile de augmentare pe train, si preprocesarile pentru test si in final o functie care sa returneze lucruri specifice unei retele neurale cum ar fi optimizatorul, learning rate scheduler-ul, criterion si o instanta a clasei care implementeaza early stopping. Pentru a rula initial foloesesc, ca pipeline de transformari, augmentarile din etapa 1. De asemenea prin incercari repetate

```python
def split_for_cross_validation(final_info: list[tuple], number_splits : int):
    validation_split = 1 / number_splits
    chunk_values = []
    for i in range(number_splits):
        validation_split = 1 / (number_splits - i)
        final_info, chunk = split_traing_data(final_info, validation_split)
        chunk_values.append(chunk)
    return chunk_values
```

(a) Imparte datele pentru cross validation

```python
def compute_cross_validation(parameters, K, data_chunks, test_info, train_transforms, test_transforms, compute_functions):
    validation_info = {"precision": [], "recall": [], "f1Score" : [], "accuracy" : []}
    info_test = {"precision": [], "recall": [], "f1Score" : [], "accuracy" : []}
    for i in range(K):
        model = Net(width=parameters["width"], expansion=parameters["expansion"])
        device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
        model = model.to(device)
        criterion, optimizer, lr_scheduler, early_stopping = compute_functions(model, parameters)
        data = create_cross_validation_data(data_chunks, test_info, i, train_transforms, test_transforms, parameters["epochs"])
        training_loss, validation_loss, training_accuracy, validation_accuracy = train_model(model, parameters["epochs"],
            data, criterion, optimizer, lr_scheduler, early_stopping)
        show_plot(training_loss, validation_loss, "Train Loss", "Validation Loss", f"Loss model{i}")
        show_plot(training_accuracy, validation_accuracy, "Train Accuracy", "Validation Accuracy", f"Accuracy model{i}")
        if (early_stopping != None):
            best_model = Net(width=parameters["width"], expansion=parameters["expansion"])
            early_stopping.load_model(best_model)
        else:
            best_model = model
        precision, recall, f1, acc, nat = test_model(best_model, data["validation"], criterion)
        validation_info = add_info(validation_info, precision, acc, recall, f1)
        precision, recall, f1, acc, nat = test_model(best_model, data["test"], criterion)
        info_test = add_info(info_test, precision, acc, recall, f1)
    return pd.DataFrame.from_dict(info_test, "columns"), pd.DataFrame.from_dict(validation_info, "columns")
```

(b) Antreneaza K modele si salveaza datele

Figure 1: Functii importante pentru pipeline-ul de antrenare

am obtinut arhitectura pe care o urmeaza reteaua mea neurala. Aceasta contine aproximativ 1603828 parametrii. Parametrii width si expansion au rolul de a determina cat de mult reteaua trebuie sa se extinde pe latime. Facand foarte flexibila modificarea arhitecturii pentru a avea mai multi sau mai putini parametrii in functie de rezultatele antrenarii. In modelul final, width=16 iar expansion=4.

2 Cerinta 2

Exista 3 situatii care trebuiesc testate independent:

1. label-urile claselor minoritare sunt duplicate, dar nu se aplica augmentari
2. label-urile claselor minoritare sunt duplicate, se aplica augmentari pe imagini
3. label-uriele nu sunt duplicate, dar functia de loss primeste ca parametru in plus un vector de weights, reprezentand gradul de importanta al unei imagini dintr-o anumita clasa.

Pentru fiecare dintre cele 3 situatii, am pastrat exact aceeasi retea neurala. Am antrenat pentru epochs = 30, batch size-ul a fost de 50, am avut learning rate de 1e-3, optimizator Adams cu parametrul weights_decay setat la 1e-4. Ca functie de augmentare a datelor de training am avut urmatorul pipeline, folosit si in etapa 1 a proiectului:

- se aplica un resize la (100, 100), am observat rezultate foarte bune cu o dimensiune mult mai mica a datelor de intrare, decat cea initiala de (512, 512), pastrandu-se in acelasi timp feature-urile relevante
- se aplica in mod aleator corectie gama pe datele de intrare, uniformizandu-le si rezolvand zgomotul aparut in procesul de obtinere al pozelor
- se face o rotire la 180 de grade cu o probabilitate de 0.5, pentru a creste capacitatea de generalizare a modelului

- se imbina operatii de rotatie, shift-are a unor pixeli, si scalare a unor portiuni din imagine, pentru a obtine date noi,
- se aplica filtrul CLAHE pentru a evidentia contururile si a scoate tumora in evidenta
- se aplica un contrast random de lumina pentru a creste diversitatea datelor
- datele sunt normalizate mean=(0.5, 0.5, 0.5), std=(0.5, 0.5, 0.5) pentru fiecare din cele 3 canale, pentru usura munca modelului.

2.1 Duplicarea clase minoritare, fara augmentari

Table 1: Rezultatele pe setul de Test, fara augmentari, duplicarea datelor

Model	**Precizie (%)**	**Recall (%)**	**F1-Score (%)**	**Acuratețe (%)**
Model 0	61.26	70.73	61.79	60.66
Model 1	69.85	71.69	69.64	70.05
Model 2	69.10	69.37	68.19	68.53
Model 3	65.70	76.54	63.70	65.23
Model 4	70.48	75.14	66.77	70.05
Media	67.28	72.69	66.02	66.90
Deviația Standard	4	3	3	4

Table 2: Rezultatele pe setul de Validation, fara augmentari, duplicarea datelor

Model	**Precizie (%)**	**Recall (%)**	**F1-Score (%)**	**Acuratețe (%)**
Model 0	73.92	83.64	71.60	73.96
Model 1	88.07	89.79	88.01	88.07
Model 2	79.63	86.72	79.11	79.70
Model 3	84.42	85.73	84.47	84.44
Model 4	88.78	88.95	88.58	88.80
Media	82.96	86.97	82.35	82.99
Deviația Standard	6	2	7	6

Rularea a durat 15 minute si 30 de secunde. Observam ca majoritatea modelelor dupa 30 de epoci incepi sa faca overfitting.

2.2 Duplicare pe clase minoritare cu augmentari

Table 3: Rezultatele pe setul de Test, cu augmentari, duplicare

Model	**Precizie (%)**	**Recall (%)**	**F1-Score (%)**	**Acuratețe (%)**
Model 0	61.26	70.73	61.79	60.66
Model 1	69.85	71.69	69.64	70.05
Model 2	69.10	69.37	68.19	68.53
Model 3	65.70	76.54	63.70	65.23
Model 4	70.48	75.14	66.77	70.05
Media	67.28	72.69	66.02	66.90
Deviația Standard	4	3	3	4

Table 4: Rezultatele pe setul de Validation, cu augmentari, duplicare

Model	**Precizie (%)**	**Recall (%)**	**F1-Score (%)**	**Acuratețe (%)**
Model 0	73.92	83.64	71.60	73.96
Model 1	88.07	89.79	88.01	88.07
Model 2	79.63	86.72	79.11	79.70
Model 3	84.42	85.73	84.47	84.44
Model 4	88.78	88.95	88.58	88.80
Media	82.96	86.97	82.35	82.99
Deviația Standard	6	2	7	6

2.3 Fara duplicare, cu augmentari

Table 5: Rezultatele pe setul de Test, cu augmentari, fara duplicare

Model	**Precizie (%)**	**Recall (%)**	**F1-Score (%)**	**Acuratețe (%)**
Model 0	62.97	60.66	58.60	60.91
Model 1	71.46	75.39	70.69	71.07
Model 2	64.92	67.80	61.15	63.45
Model 3	68.14	70.12	65.79	67.77
Model 4	65.99	65.90	65.74	65.48
Media	66.69	67.97	64.39	65.74
Deviația Standard	3	5	5	4

Table 6: Rezultatele pe setul de Validation, cu augmentari, fara duplicare

Model	**Precizie (%)**	**Recall (%)**	**F1-Score (%)**	**Acuratețe (%)**
Model 0	78.86	82.33	76.68	78.09
Model 1	85.62	88.69	86.69	88.24
Model 2	85.89	89.90	86.48	86.68
Model 3	85.56	85.55	85.52	86.03
Model 4	79.28	83.80	80.48	81.44
Media	83.04	86.06	83.17	84.09
Deviația Standard	4	3	4	4

3 Cerinta 3 - Transformarile din Monai

```python
first_augmentation_pipeline = Compose([
    Resize(spatial_size= SIZE),
    RandGaussianSharpen(),
    RandAxisFlip(prob=0.3),
    RandSpatialCrop(roi_size=SIZE, random_center=True, random_size=False),
    RandRotate(range_x=(-50, 50),  prob=0.5, keep_size=True),
    RandAdjustContrast(prob=0.8),
    ScaleIntensity(minv=0.0, maxv=1.0)
])

second_augmentation_pipeline = Compose([
    Resize(spatial_size= SIZE),
    Rand2DElastic(spacing=(5, 10), magnitude_range=(1, 2), prob=0.5, rotate_range=(0, 0), shear_range=(0.02, 0.02),
                  translate_range=(10, 10), scale_range=(0.1, 0.1), spatial_size=SIZE, mode="bilinear", padding_mode="reflection"),
    RandHistogramShift(prob=0.8),
    ThresholdIntensity(threshold=20),
    RandGaussianSmooth(sigma_x=(0.5, 1.5), sigma_y=(0.5, 1.5), prob=0.8),
    RandRotate(range_x=(-50, 50),  prob=0.5, keep_size=True),
    ScaleIntensity(minv=0.0, maxv=1.0)
])


third_augmentation_pipeline = Compose([
    Resize(spatial_size= SIZE),
    RandGaussianNoise(mean=0.0, std=0.1, prob=0.8),
    RandShiftIntensity(offsets=(-20, 20), prob=0.3, safe=True),
    RandAxisFlip(prob=0.3),
    RandStdShiftIntensity(factors=0.8),
    RandAffine(prob=0.2),
    ScaleIntensity(minv=0.0, maxv=1.0)
])

test_augmentation_pipeline = Compose([
    Resize(spatial_size= SIZE),
    ScaleIntensity(minv=0.0, maxv=1.0)
])
```

Figure 17: Transformari Monai Task 3

Am implementat 3 tipuri de transformari diferite si de asemenea am implementat un pipeline scurt si pentru test, in care imaginea sufera un resize si o scalare in intervalul [0, 1].

3.1 Transformarea 1

Model	Precision	Recall	F1-Score	Accuracy
Model 0	63.86	73.65	62.36	66.50
Model 1	55.87	69.89	54.14	55.84
Model 2	60.93	76.39	57.14	60.91
Model 3	68.67	68.57	65.91	68.53
Model 4	62.17	68.94	60.69	63.45
Mean	62.30	71.49	60.05	63.05
Standard Deviation	4.64	3.40	4.57	4.97

Model	Precision	Recall	F1-Score	Accuracy
Model 0	90.74	91.30	0.9087	90.75
Model 1	88.45	88.91	0.8804	88.45
Model 2	87.48	88.92	0.8694	87.48
Model 3	89.53	89.97	0.8935	89.56
Model 4	90.33	91.20	0.9051	90.32
Mean	89.31	0.9006	89.14	89.31
Standard Deviation	1.34	1.17	1.65	1.35

3.2 Transformarea 2

Table 7: Rezultatele pe setul de Test

Model	Precizie (%)	Recall (%)	F1-Score (%)	Acuratețe (%)
Model 0	62.27	71.48	61.98	63.71
Model 1	67.81	74.66	62.72	66.75
Model 2	62.01	69.14	58.27	62.69
Model 3	46.91	68.52	45.09	46.45
Model 4	65.29	67.21	61.43	63.96
Media	60.86	70.20	57.90	60.71
Deviația Standard	8	3	7	8

Table 8: Rezultatele pe setul de Validation

Model	Precizie (%)	Recall (%)	F1-Score (%)	Acuratețe (%)
Model 0	87.91	88.73	87.82	87.92
Model 1	84.47	86.57	84.12	84.47
Model 2	88.43	89.15	88.35	88.43
Model 3	71.99	80.40	65.85	72.11
Model 4	87.82	88.30	87.52	87.86
Media	84.12	86.63	82.73	84.16
Deviația Standard	7	4	10	7

3.3 Transformarea 3

Table 9: Rezultatele pe setul de Test

Model	Precizie (%)	Recall (%)	F1-Score (%)	Acuratețe (%)
Model 0	56.96	74.06	53.41	59.90
Model 1	49.63	78.03	49.28	51.78
Model 2	64.78	66.29	65.41	64.21
Model 3	65.20	77.17	63.50	66.75
Model 4	65.62	78.89	63.51	66.50
Media	60.44	74.89	59.02	61.83
Deviația Standard	7	5	7	6

Table 10: Rezultatele pe setul de Validation

Model	Precizie (%)	Recall (%)	F1-Score (%)	Acuratețe (%)
Model 0	86.58	88.47	86.78	86.60
Model 1	79.55	87.98	80.20	79.55
Model 2	81.17	87.70	81.43	81.21
Model 3	91.63	91.70	91.54	91.65
Model 4	89.35	89.88	89.22	89.37
Media	85.65	89.14	85.83	85.68
Deviația Standard	5	2	5	5

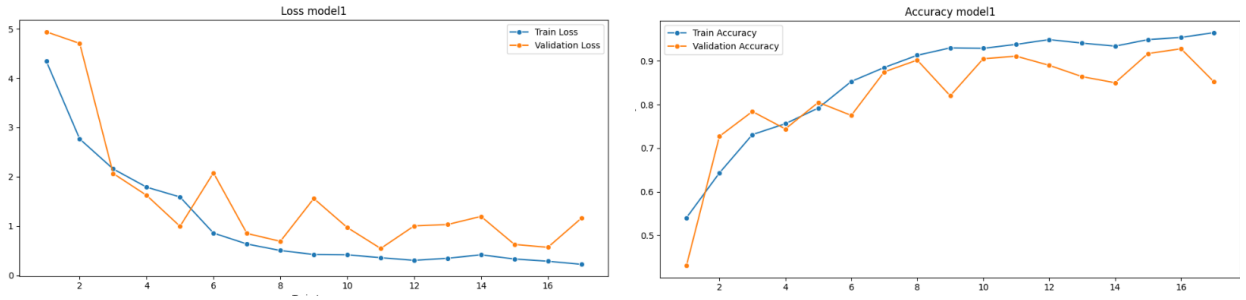
Cel mai bun pipeline de transformari este primul atat pe setul de test cat si pe cel de validare avand cele mai bune rezultate. Un posibil motiv ar fi faptul ca functia gaussiana de sharpen scoate in evidenta contururile deja existente, scotand in evidenta tesutul tumoral. De asemenea restul transformarilor din acest pipeline doar modifica orientarea si intensitatea luminoasa, dar nu forma imaginilor de tip MRI, sau nu scad acuratetea precum celelalte.

4 Cerinta 4

Pentru prima transformare, care are cele bune rezultate in general, obsercam ca modelul 1 are cele mai proasta rezultate pentru toate metricile. Acuratetea fiiind de 55%, f1-score: 54%, recall : 69%, precision 55%. Iar timpul de antrenament a fost de 31 de minute. Vom aplica pe rand cele 2 metode pentru a imbunatati acuratetea pe fold-ul 1.

4.1 Implementare Early Stopping

Prima data am aplicat Early Stopping cu o rata de oprire la 5 epoci distanta.



(a) Loss model1 training si validation

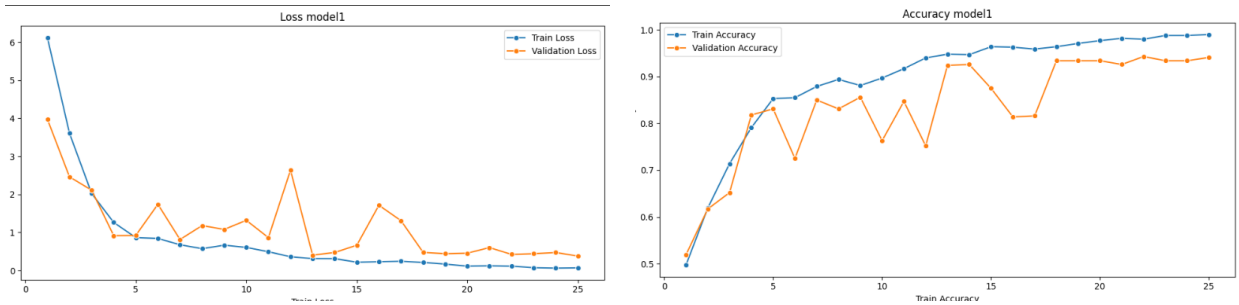
(b) Acuratete model1 training si validation

Figure 33: Loss si acuratete pe modelul 1 dupa aplicarea strategiei de early stopping

In urma rularii modelul se opreste dupa 18 epoci, in loc de 30 precum era inainte. Iar modificarile pe care le-am obtinut sunt: precizie: 63%, recall: 74%, f1 score: 61%, 64% acuratete. Aceste rezultate fiind obtinute pe datele de test. Acum timpul de antrenament a fost 4 minute.

4.2 Implemenetare Learning Rate Scheduler

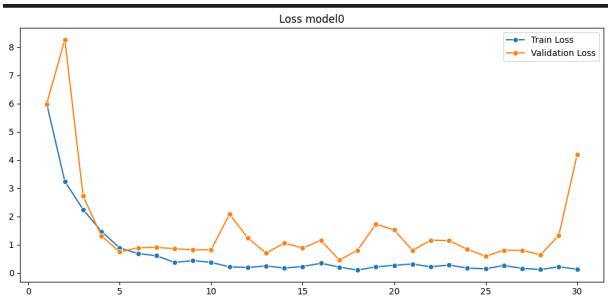
Pentru a antrena am folosit ReduceLROnPlateau, care tinea evidenta loss-ului pentru validare. Daca loss-ul respectiv nu se imbunatatea pentru un numar de 3 epoci, learning rate-ul devenea lr = lr * 0.6. Initial am antrenat pentru 30 de epoci ca inainte, dar am observat ca dupa epoca 12, valoarea de loss pentru training si accuracy se stabilizeaza, iar valoarea nu se mai modifica deloc, in aceeasi situatie aflandu-se si acuratetea. Din aceasta cauza am redus numarul de epoci la 25. Observand ca in continuare obtin rezultate bune. Acuratetea a deveni: 71%, f1 score : 68%, recall 80%, precision 70%. Si a durat 5 minute jumatate.



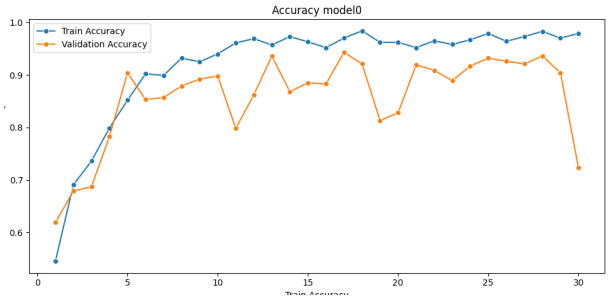
(a) Loss model1 training si validation

(b) Acuratete model1 training si validation

Figure 34: Loss si acuratete pe modelul 1 dupa aplicarea strategiei de early stopping

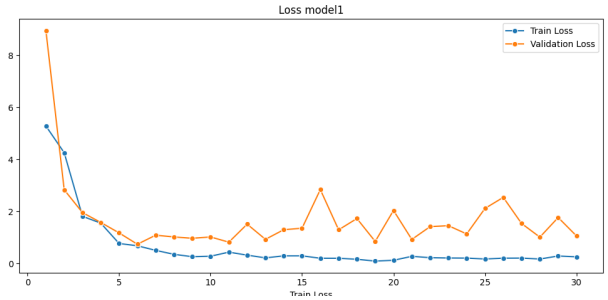


(a) Loss model0 training si validation

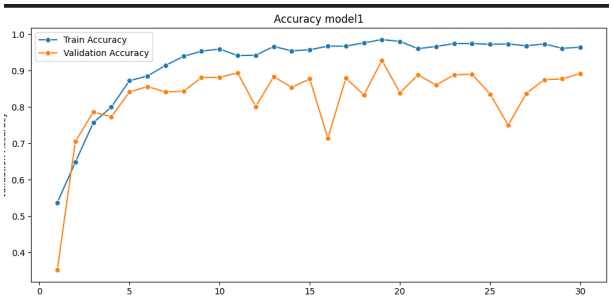


(b) Acuratete model0 training si validation

Figure 2: Loss si acuratete pe modelul 0, fara augmentari, duplicare

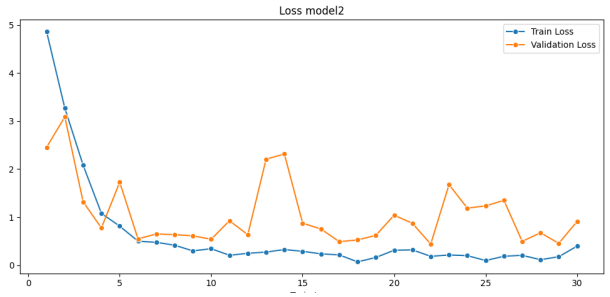


(a) Loss model1 training si validation

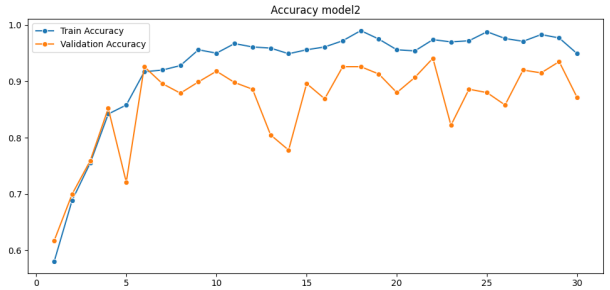


(b) Acuratete model1 training si validation

Figure 3: Loss si acuratete pe modelul 1, fara augmentari, duplicare

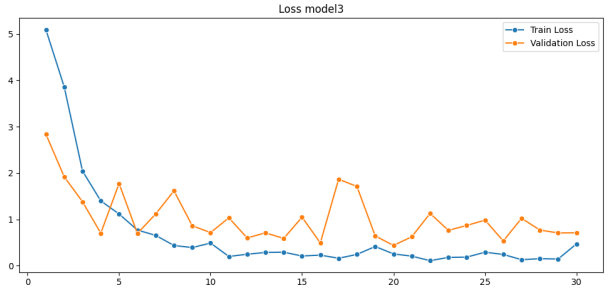


(a) Loss model2 training si validation

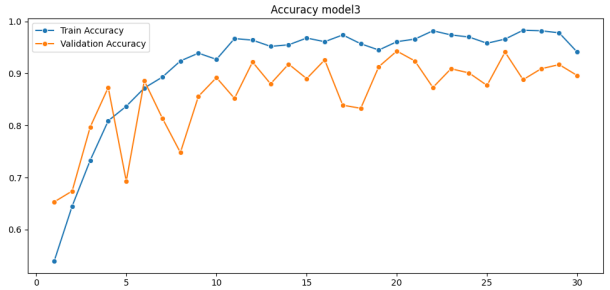


(b) Acuratete model2 training si validation

Figure 4: Loss si acuratete pe modelul 2, fara augmentari, duplicare

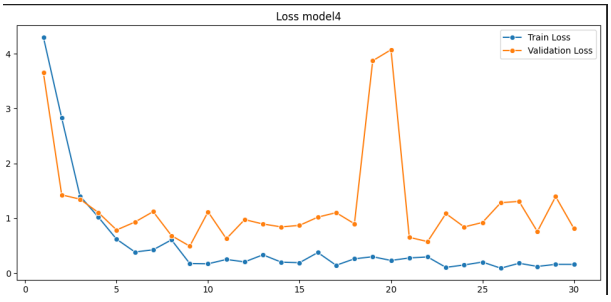


(a) Loss model3 training si validation

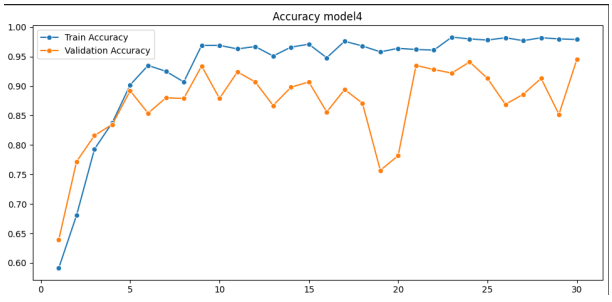


(b) Acuratete model3 training si validation

Figure 5: Loss si acuratete pe modelul 3, fara augmentari, duplicare

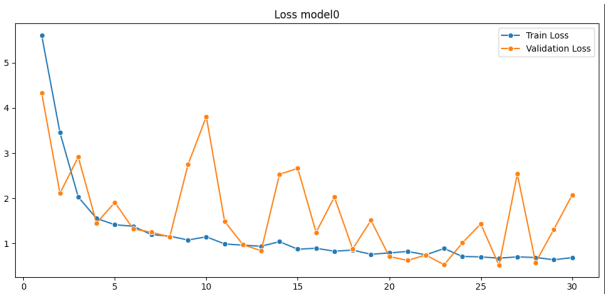


(a) Loss model4 training si validation

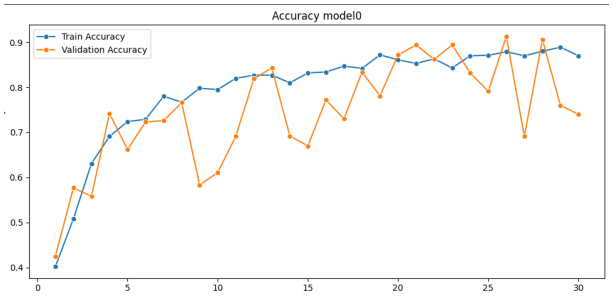


(b) Acuratete model4 training si validation

Figure 6: Loss si acuratete pe modelul 4, fara augmentari, duplicare

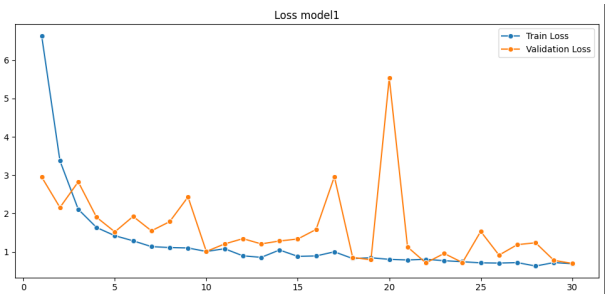


(a) Loss model0 training si validation

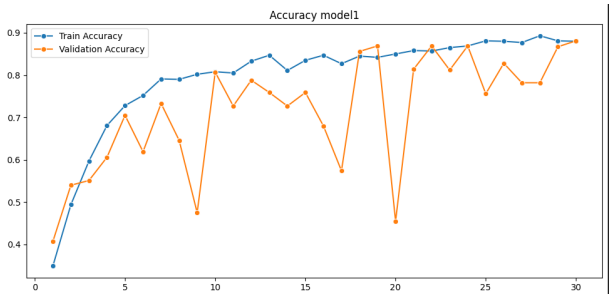


(b) Acuratete model0 training si validation

Figure 7: Loss si acuratete pe modelul 0, cu augmentari, duplicare

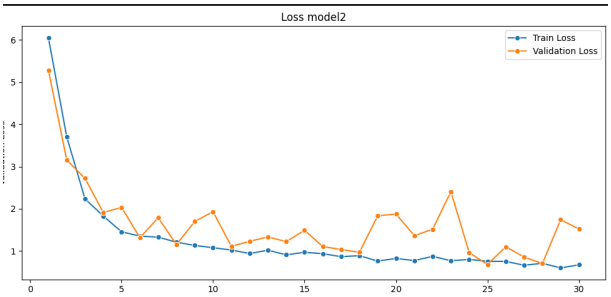


(a) Loss model1 training si validation

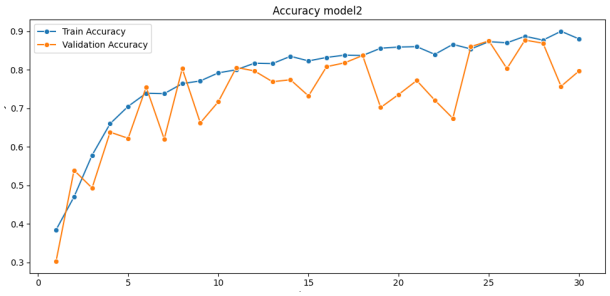


(b) Acuratete model1 training si validation

Figure 8: Loss si acuratete pe modelul 1, cu augmentari, duplicare

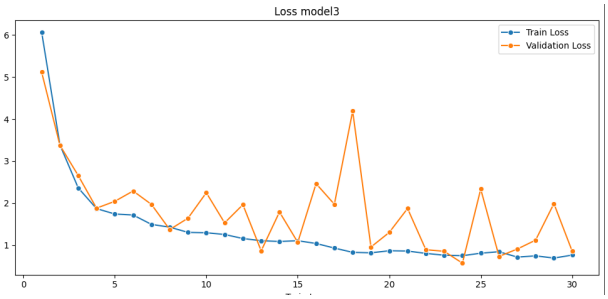


(a) Loss model2 training si validation

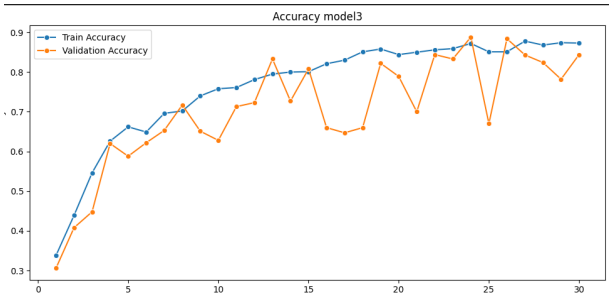


(b) Acuratete model2 training si validation

Figure 9: Loss si acuratete pe modelul 2, cu augmentari, duplicare

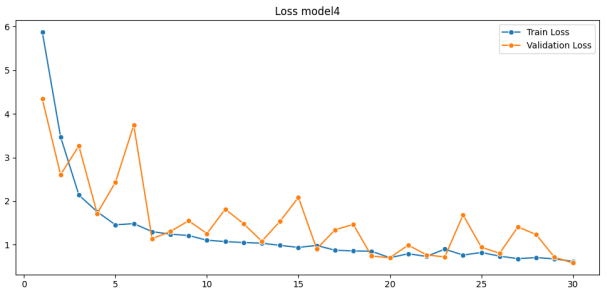


(a) Loss model3 training si validation

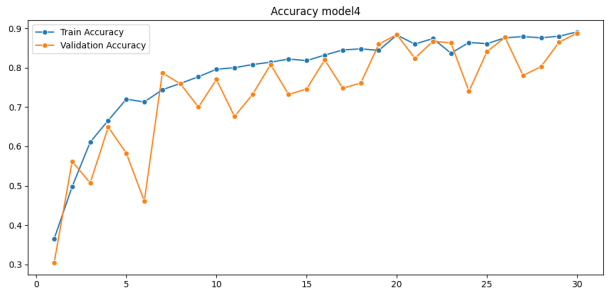


(b) Acuratete model3 training si validation

Figure 10: Loss si acuratete pe modelul 3, cu augmentari, duplicare

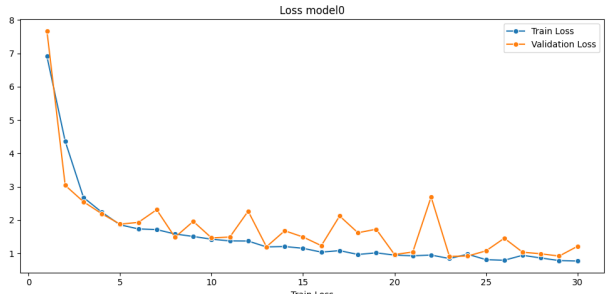


(a) Loss model4 training si validation

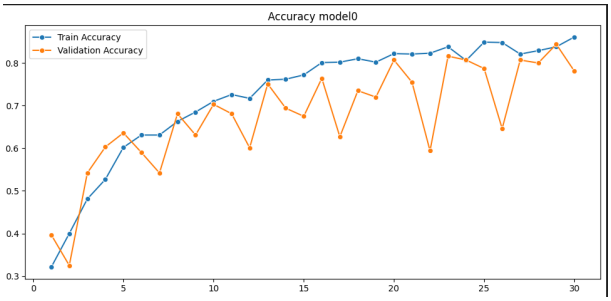


(b) Acuratete model4 training si validation

Figure 11: Loss si acuratete pe modelul 4, cu augmentari, duplicare

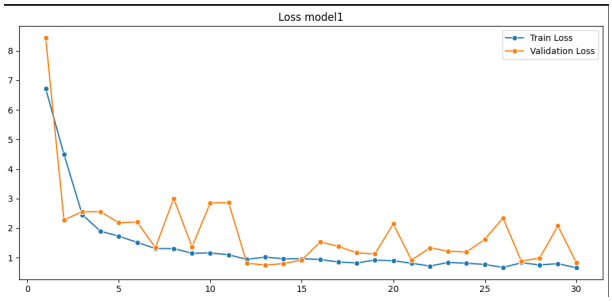


(a) Loss model0 training si validation

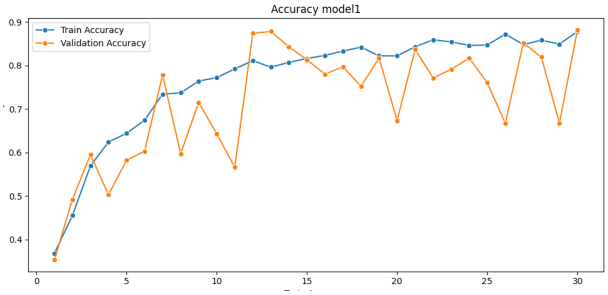


(b) Acuratete model0 training si validation

Figure 12: Loss si acuratete pe modelul 0, cu augmentari, fara duplicare

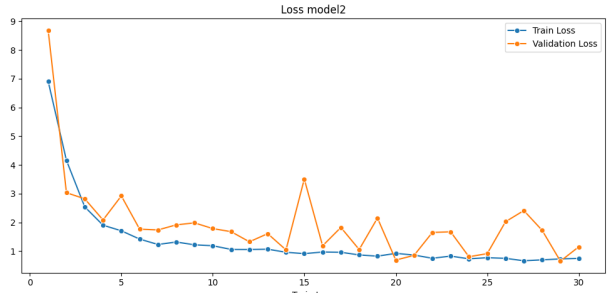


(a) Loss model1 training si validation

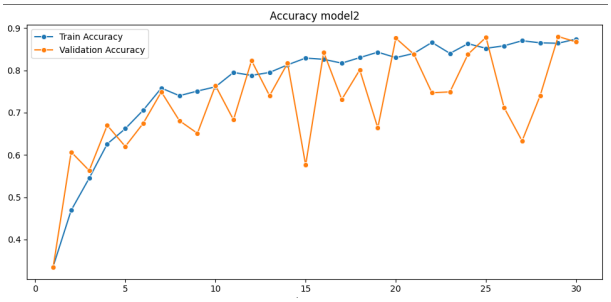


(b) Acuratete model1 training si validation

Figure 13: Loss si acuratete pe modelul 1, cu augmentari, fara duplicare

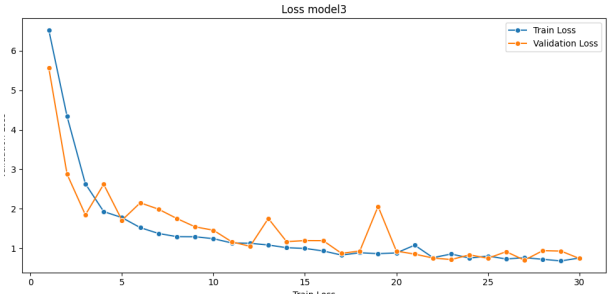


(a) Loss model2 training si validation

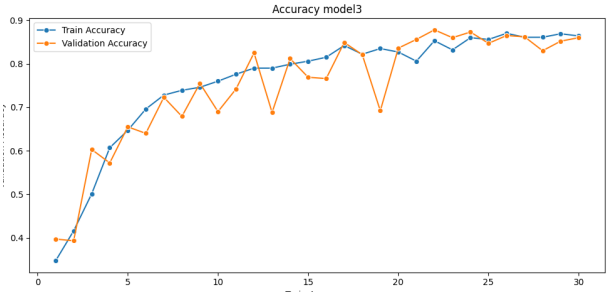


(b) Acuratete model2 training si validation

Figure 14: Loss si acuratete pe modelul 2, cu augmentari, fara duplicare

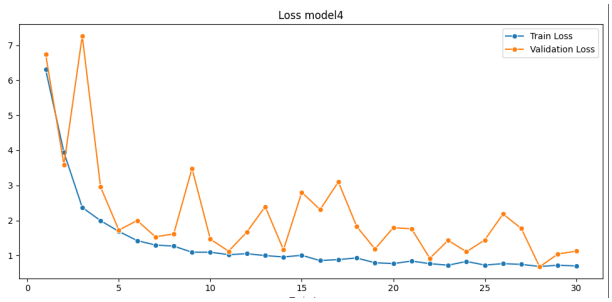


(a) Loss model3 training si validation

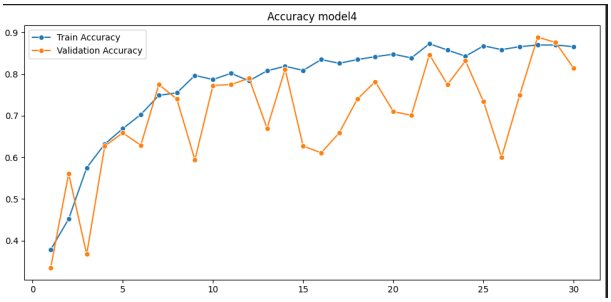


(b) Acuratete model3 training si validation

Figure 15: Loss si acuratete pe modelul 3, cu augmentari, fara duplicare

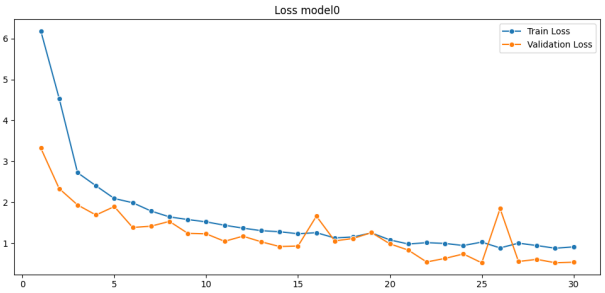


(a) Loss model4 training si validation

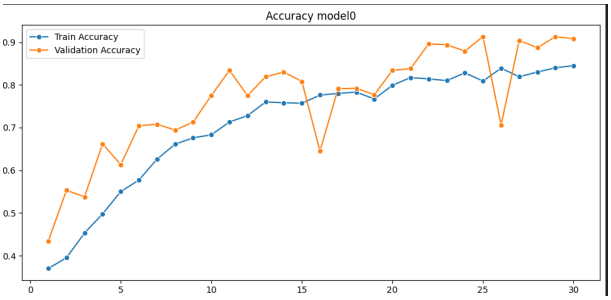


(b) Acuratete model4 training si validation

Figure 16: Loss si acuratete pe modelul 4, cu augmentari, fara duplicare

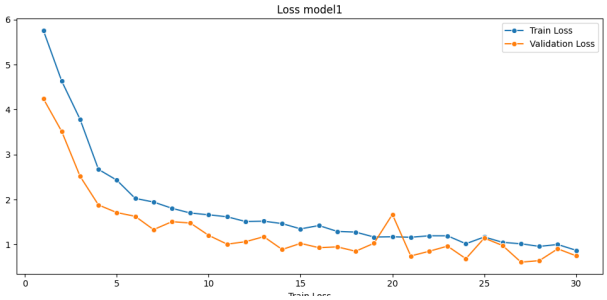


(a) Loss model0 training si validation

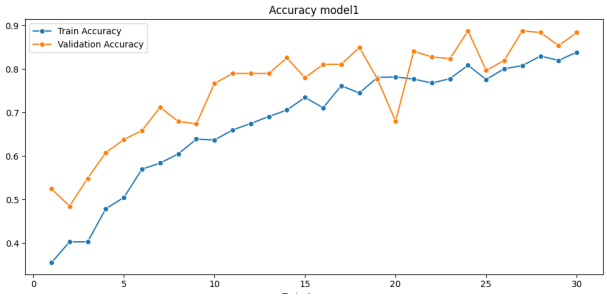


(b) Acuratete model0 training si validation

Figure 18: Loss si acuratete pe modelul 0 primul set de augmentari MONAI

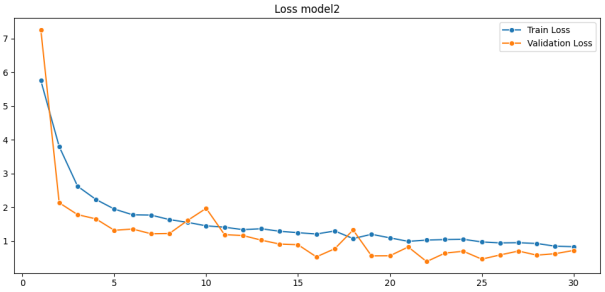


(a) Loss model1 training si validation

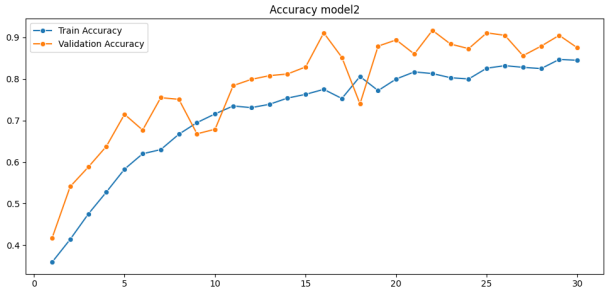


(b) Acuratete model1 training si validation

Figure 19: Loss si acuratete pe modelul 1 primul set de augmentari MONAI

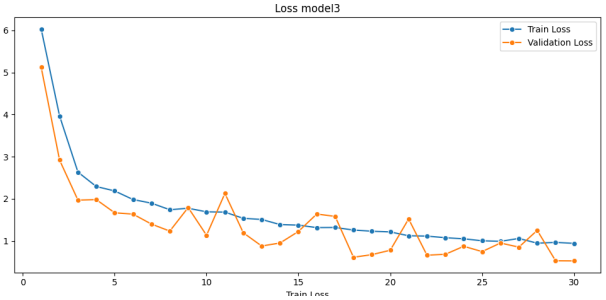


(a) Loss model2 training si validation

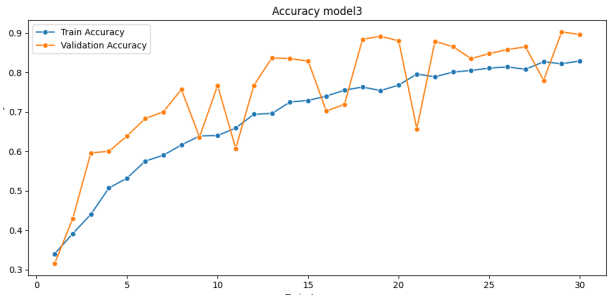


(b) Acuratete model2 training si validation

Figure 20: Loss si acuratete pe modelul 2 primul set de augmentari MONAI

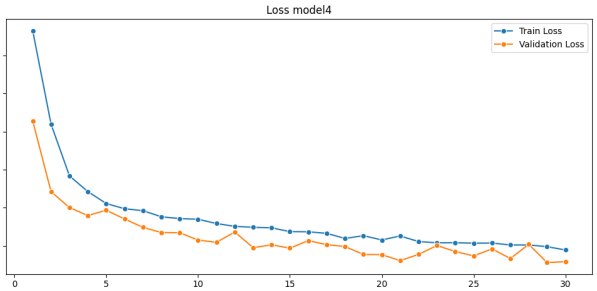


(a) Loss model3 training si validation

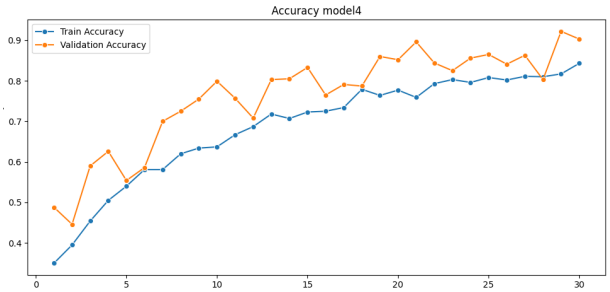


(b) Acuratete model3 training si validation

Figure 21: Loss si acuratete pe modelul 3 primul set de augmentari MONAI

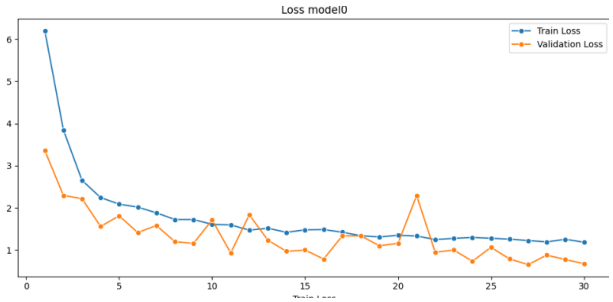


(a) Loss model4 training si validation

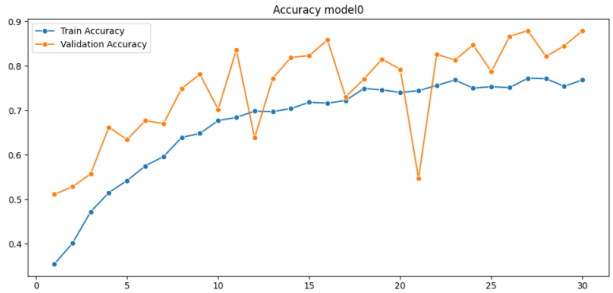


(b) Acuratete model4 training si validation

Figure 22: Loss si acuratete pe modelul 4 primul set de augmentari MONAI

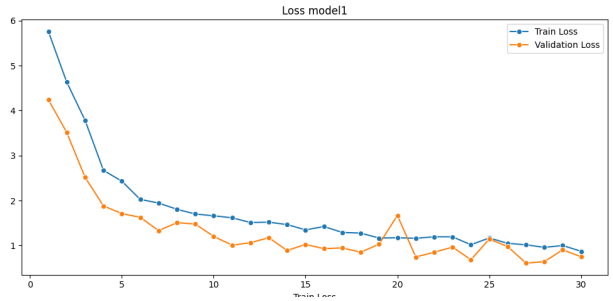


(a) Loss model0 training si validation

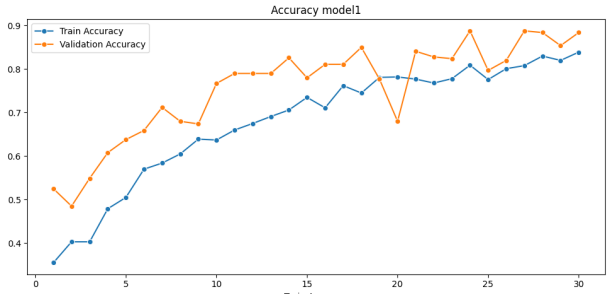


(b) Acuratete model0 training si validation

Figure 23: Loss si acuratete pe modelul 0, al doilea set de augmentari MONAI

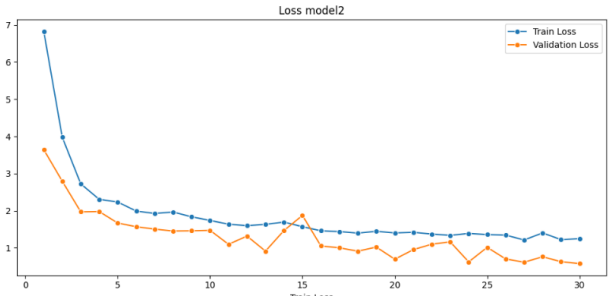


(a) Loss model1 training si validation

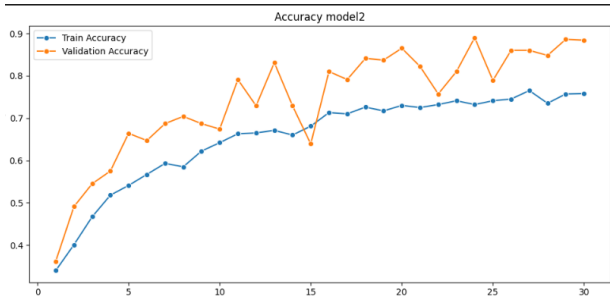


(b) Acuratete model1 training si validation

Figure 24: Loss si acuratete pe modelul 1, al doilea set de augmentari MONAI

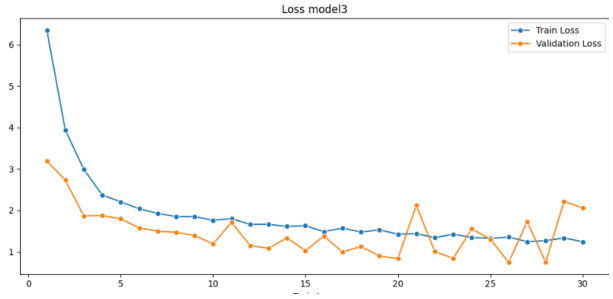


(a) Loss model2 training si validation



(b) Acuratete model2 training si validation

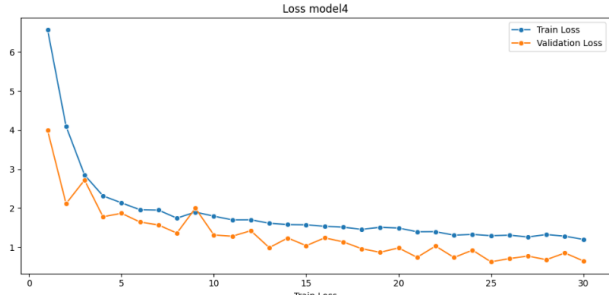
Figure 25: Loss si acuratete pe modelul 2, al doilea set de augmentari MONAI



(a) Loss model3 training si validation

(b) Acuratete model3 training si validation

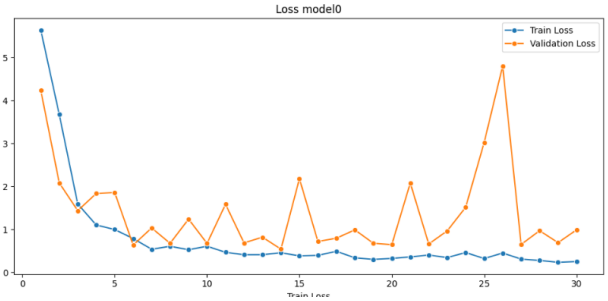
Figure 26: Loss si acuratete pe modelul 3, al doilea set de augmentari MONAI



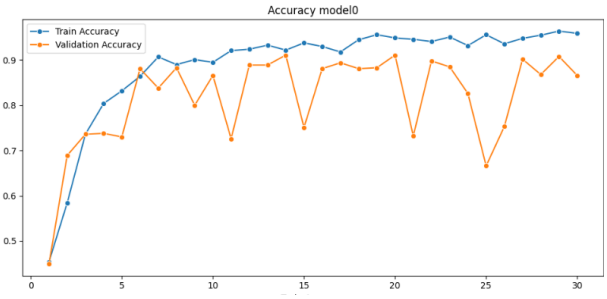
(a) Loss model4 training si validation

(b) Acuratete model4 training si validation

Figure 27: Loss si acuratete pe modelul 4, al doilea set de augmentari MONAI

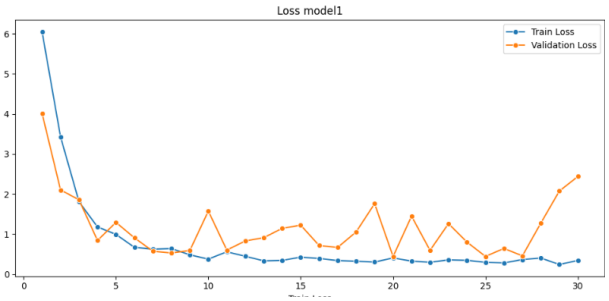


(a) Loss model0 training si validation

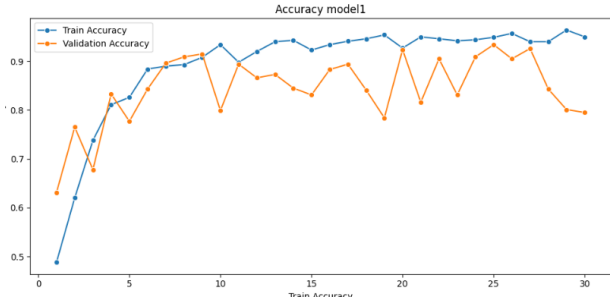


(b) Acuratete model0 training si validation

Figure 28: Loss si acuratete pe modelul 0, al treilea set de augmentari MONAI

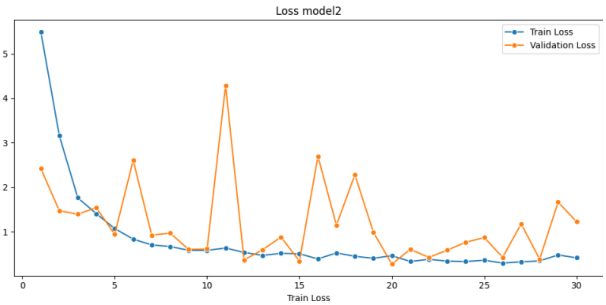


(a) Loss model1 training si validation

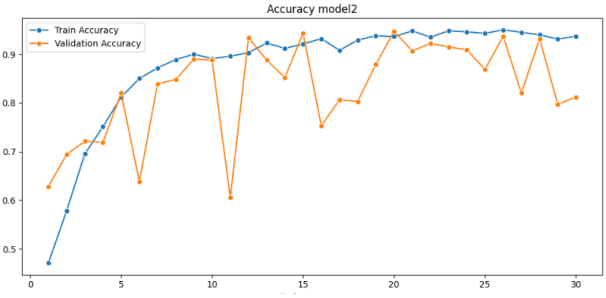


(b) Acuratete model1 training si validation

Figure 29: Loss si acuratete pe modelul 1, al treilea set de augmentari MONAI

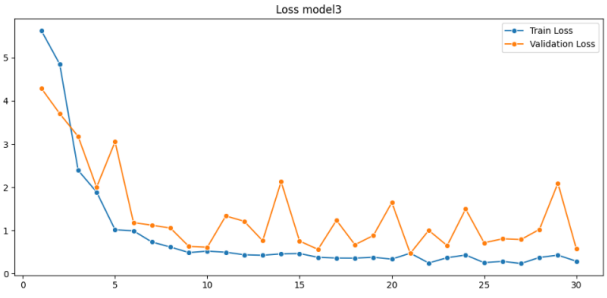


(a) Loss model2 training si validation

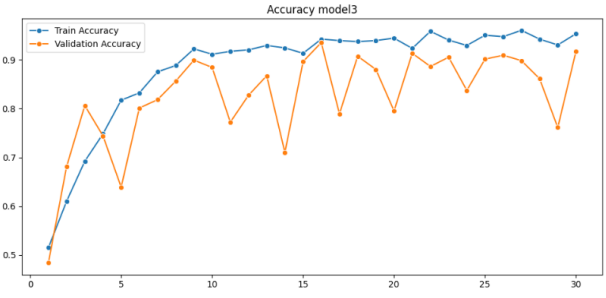


(b) Acuratete model2 training si validation

Figure 30: Loss si acuratete pe modelul 2, al treilea set de augmentari MONAI

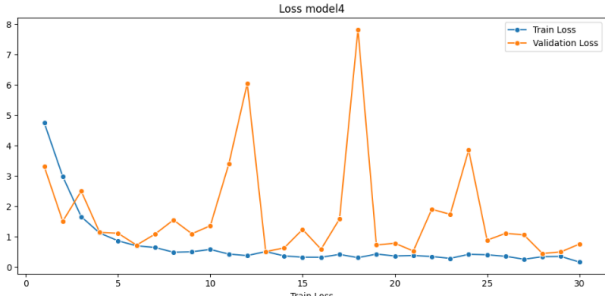


(a) Loss model3 training si validation

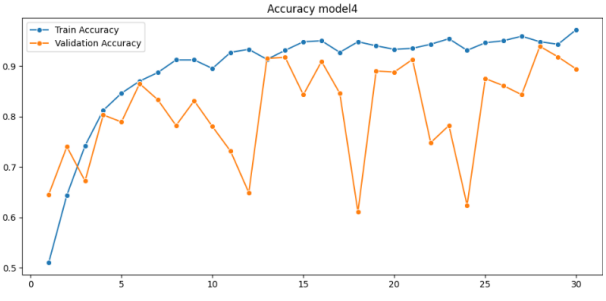


(b) Acuratete model3 training si validation

Figure 31: Loss si acuratete pe modelul 3, al treilea set de augmentari MONAI



(a) Loss model4 training si validation



(b) Acuratete model4 training si validation

Figure 32: Loss si acuratete pe modelul 4, al treilea set de augmentari MONAI