Verificarea Semnaturii utilizand TensorFlow

Ciobanu Mihaela Lavinia
Facultatea de Electronică,
Telecomunicații și Tehnologii
Informaționale
Universitatea Politehnica Timișoara
Timișoara, România
mihaela.ciobanu@student.upt.ro

***Abstract*—Cercetările actuale investighează o varietate de metode pentru a distinge semnăturile autentice de cele falsificate cu o precizie ridicată. Scopul lucrării de față este de a dezvolta un sistem nou, fiabil, pentru asigurarea securității și autenticității tranzacțiilor în diverse domenii, cum ar fi plățile mobile, cloud computing și gestionarea documentelor digitale. Sistemul propus este construit utilizând TensorFlow, o platformă recunoscută pentru capacitățile sale în domeniul învățării profunde. În centrul acestui sistem se află o Rețea Neurală Convoluțională (CNN), care poate verifica cu exactitate caracteristicile specifice unei semnături autentice. Performanța sistemului este măsurată prin doi indicatori principali: Rata de Respingere Falsă (FRR) și Rata de Acceptare Falsă (FAR). În cadrul testelor, sistemul a obținut valori de 5% atât pentru FRR, cât și pentru FAR, iar acuratețea generală a fost de 90%.**

Cuvinte cheie: Verificarea semnăturilor, Învățare profundă, Rețea Neurală Convoluțională (CNN), TensorFlow, Securitatea tranzacțiilor, Autenticitate, Rata de Respingere Falsă (FRR), Rata de Acceptare Falsă (FAR), Plăți mobile, Cloud computing, Gestionarea documentelor digitale

I. Introducere

Semnăturile de mână reprezintă o metodă biometrică neinvazivă și sunt frecvent utilizate pentru a confirma autenticitatea documentelor. În verificarea semnăturilor se identifică două probleme esențiale: identificarea și autentificarea semnăturii. În cazul identificării, procesul urmărește să stabilească identitatea semnatarului unui document. În cazul autentificării, se evaluează gradul de similaritate dintre o semnătură de test și una de referință pentru a determina dacă semnătura este autentică sau falsificată. Aceste procese sunt adesea complicate de variabilitatea intrapersonală și interpersonală a semnăturilor, influențată de factori precum spațiul disponibil pentru semnare, tipul instrumentului de scris, starea psihologică și fizică a semnatarului, precum și alte condiții particulare.[3]

În detectarea falsificării semnăturilor de mână, existența unui sistem care poate distinge eficient între semnăturile autentice și cele contrafăcute este esențială pentru a preveni cazurile de falsificare a identității, în special în domeniul bancar și în alte instituții care se bazează pe semnătura manuală pentru verificarea identității. Un astfel de sistem poate proteja instituțiile de pierderi semnificative de informații și resurse de valoare, fiind util în mediile care necesită verificarea constantă a autenticității semnăturilor clienților. [1]

Implementarea unui sistem bazat pe tehnici de învățare automată și profundă, cum ar fi Rețelele Neuronale Convoluționale (CNN), poate facilita identificarea rapidă și precisă a semnăturilor autentice în comparație cu alte soluții, cum ar fi Rețelele Neuronale Artificiale (ANN) sau Rețelele Neuronale Recurente (RNN). Acest tip de sistem asigură un echilibru optim între eficiență și acuratețe în procesul de verificare, oferind o abordare modernă și robustă în protecția împotriva falsificării documentelor.[2]

Lucrarea este organizată în cinci secțiuni: a doua secțiune prezintă fundamentele teoretice, a treia secțiune descrie implementarea, a patra secțiune expune rezultatele, iar concluziile sunt oferite în secțiunea finală.

II. Fundamente Teoretice

A. *Rețele Neuronale Convoluționale (CNN)*

Rețelele neuronale convoluționale (CNN) sunt similare rețelelor neuronale standard, având cel puțin un strat de convoluție urmat de cel puțin un strat complet conectat, asemănător unei rețele neuronale tradiționale. CNN-urile sunt alcătuite din neuroni cu ponderi variabile, care sunt ajustate în timpul fazei de antrenare. Unul dintre avantajele CNN-urilor este necesitatea minimă de preprocesare a datelor de intrare, acestea fiind formate din mai multe perceptroni stratificați. Arhitectura CNN este concepută astfel încât să profite la maximum de semnalele bidimensionale de intrare, făcându-le ideale pentru clasificarea imaginilor și procesarea limbajului natural [1]. In figura 1 este prezentată arhitectura de bază a unei rețele neuronale convoluționale (CNN).

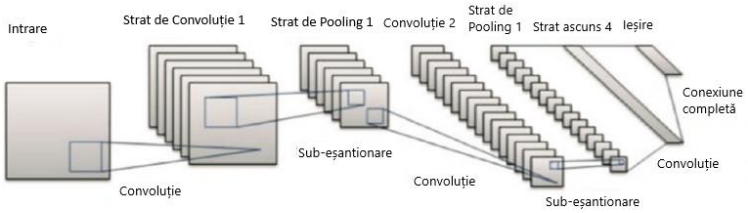

Figura 1. Arhitectura de bază a CNN [1]

B. *TensorFlow*

TensorFlow [4] este o bibliotecă open-source dezvoltată de Google, destinată să simplifice implementarea și utilizarea rețelelor neuronale complexe pentru profesioniști și cercetători.

TensorFlow utilizează matrice multidimensionale, cunoscute sub numele de tensori, pentru calculul eficient al expresiilor matematice. Programarea devine mai simplă în

TensorFlow, deoarece același cod poate fi rulat atât pe CPU, cât și pe GPU. Suportă arhitecturi cu mai multe GPU-uri, ceea ce accelerează semnificativ antrenarea rețelelor prin utilizarea capacității de procesare paralelă a unităților moderne GPU. TensorFlow este compatibil cu extensia CUDA [5] pentru calculul pe GPU [1].

C. *Memoria pe Termen Lung și Scurt (LSTM)*

LSTM-urile sunt un tip de rețele neuronale recurente (RNN) proiectate pentru a păstra informații pe termen lung și scurt în datele secvențiale. În verificarea semnăturilor, LSTM-urile sunt folosite pentru a captura relațiile temporale și secvențiale între trăsăturile extrase din imagini de către CNN. Celulele LSTM sunt compuse din trei porți principale: poarta de uitare, poarta de intrare și poarta de ieșire, care permit modelului să stocheze și să actualizeze informațiile pe mai mulți pași de timp. Această abilitate de a menține contextul secvențial este esențială pentru recunoașterea secvențială a literelor și pentru interpretarea corectă a întregii semnături [6].

D. *Funcția de Pierdere CTC (Connectionist Temporal Classification)*

Funcția de pierdere CTC este o metodă specifică pentru antrenarea rețelelor neuronale care trebuie să recunoască secvențe de lungime variabilă fără a necesita o aliniere exactă între intrare și ieșire. CTC permite recunoașterea secvențelor de caractere din semnături prin generarea unei probabilități pentru fiecare caracter la fiecare pas de timp și prin utilizarea unei funcții de scor care maximizează probabilitatea secvenței corecte. Formula CTC se poate exprima ca:

$$P(y|x) = \sum_{z \in B^{-1}(y)} P(z|x) \quad (1)$$

unde $P(y|x)$ este probabilitatea secvenței țintă y, x este secvența de intrare, și $B^{-1}(y)$ este mulțimea tuturor secvențelor de lungime variabilă care pot corespunde ieșirii y [7]

E. *Rata de Eroare pe Caracter (CER)*

CER este o metrică de evaluare utilizată pentru a măsura acuratețea recunoașterii textului la nivel de caracter. Este definită ca raportul dintre numărul de caractere incorecte și numărul total de caractere din secvența de referință. Formula CER este dată de:

$$CER = \frac{S+D+I}{N} \quad (2)$$

unde S reprezintă numărul de substituiri, D numărul de ștergeri, I numărul de inserții, și N numărul total de caractere din secvența de referință. CER este o metrică importantă în evaluarea modelelor de OCR aplicate pe semnături [8].

F. *Rata de Eroare pe Cuvânt (WER)*

WER este similară cu CER, dar măsoară acuratețea la nivel de cuvânt. Formula WER este:

$$WER = \frac{S+D+I}{N} \quad (3)$$

unde S, D, și I sunt la fel ca în CER, dar calculate la nivel de cuvânt, iar N este numărul total de cuvinte în secvența de referință. WER este folosită pentru a evalua performanța unui sistem de recunoaștere a semnăturilor la nivel de cuvinte complete [9].

G. *Recunoaștere Optică a Caracterelor (OCR)*

OCR este procesul prin care imaginile de text scris de mână sau tipărit sunt transformate în text digital, recunoscut de computer. În verificarea semnăturilor, OCR-ul este esențial pentru a extrage și interpreta caracteristicile literelor și pentru a converti semnăturile în formate care pot fi utilizate în autentificarea automată. Prin utilizarea CNN-urilor și LSTM-urilor integrate într-un sistem OCR, se obține o recunoaștere precisă și rapidă a caracterelor din semnături [9][10].

III. Implementare

Această secțiune detaliază implementarea unui sistem de verificare a semnăturilor utilizând o arhitectură de rețea neuronală convoluțională (CNN). Sistemul a fost dezvoltat folosind baza de date CEDAR, care conține semnături autentice și falsificate, reprezentate binar prin valorile 0 și 1.

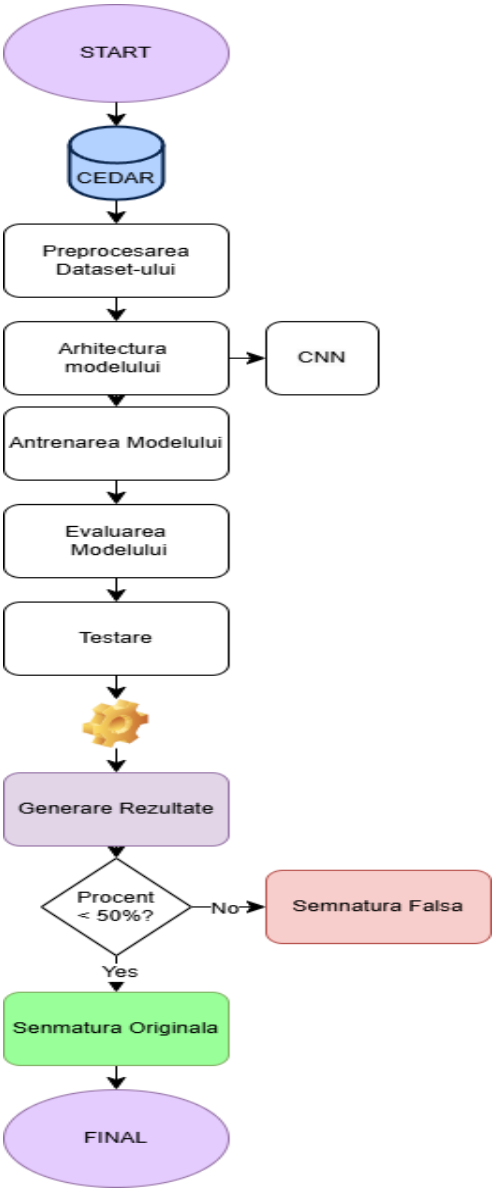

Figura 2. Algoritm verificare semnatura

A. Baza de date folosita

Datele din baza de date CEDAR Signature au fost colectate de la 55 de indivizi. Fiecare participant a furnizat 24 de semnături autentice. În plus, fiecare individ a fost rugat să falsifice semnăturile a trei alți participanți, realizând opt falsuri pentru fiecare, ceea ce a dus la un total de 1.320 de semnături falsificate. Aceste semnături au fost scanate la o rezoluție de 300 dpi în tonuri de gri și ulterior binarizate utilizând histograme de tonuri de gri. Procesarea imaginilor a inclus, de asemenea, eliminarea zgomotului de tip "sare și piper" și normalizarea înclinării.

Baza de date este organizată în două directoare principale:

- full_forg: Conține 1.320 de semnături falsificate, cu câte 24 de falsuri pentru fiecare dintre cei 55 de semnatari.
- full_org: Conține 1.320 de semnături autentice, cu câte 24 de semnături originale pentru fiecare dintre cei 55 de semnatari.

Baza de date CEDAR Signature a fost utilizată în numeroase studii pentru antrenarea și testarea algoritmilor de verificare a semnăturilor. De exemplu, în lucrarea "Offline Signature Verification Using CNN and SVM Classifier", autorii au folosit această bază de date pentru a evalua performanța unui sistem de verificare a semnăturilor bazat pe rețele neuronale convoluționale și mașini cu vectori de suport .

Baza de date CEDAR Signature este disponibilă pentru cercetare necomercială. Pentru a obține acces, este necesar să contactați Centrul de Excelență pentru Analiza și Recunoașterea Documentelor (CEDAR) de la Universitatea din Buffalo.

Utilizarea acestei baze de date standardizate a permis compararea eficientă a diferitelor metode de verificare a semnăturilor și a contribuit semnificativ la avansarea cercetării în acest domeniu. De exemplu, în studiul "Handwritten Signature Verification via Deep Sparse Coding Architecture", baza de date CEDAR a fost utilizată pentru a demonstra eficiența unei noi arhitecturi de codare rară profundă în verificarea semnăturilor.[11]

B. Preprocesarea Dataset-ului

Preprocesarea este esențială pentru asigurarea calității modelului antrenat. Imaginile semnăturilor sunt convertite în tonuri de gri și binarizate pentru a evidenția trăsăturile distinctive ale semnăturii. Fiecare imagine este redimensionată la dimensiuni standard, de exemplu, 225x155 pixeli, pentru a asigura uniformitatea în timpul antrenării. Valorile pixelilor sunt normalizate, accelerând procesul de învățare și îmbunătățind convergența modelului.

C. Arhitectura Modelului

Arhitectura modelului utilizează o rețea neuronală convoluțională (CNN) pentru a extrage caracteristici esențiale ale semnăturii, precum contururi și structuri specifice. Straturile convoluționale permit identificarea trăsăturilor relevante, facilitând diferențierea între semnăturile autentice și cele falsificate. Funcția de activare ReLU este utilizată pentru a introduce non-linearități, iar straturile de pooling reduc dimensiunile caracteristicilor extrase, menținând informațiile esențiale.

D. Antrenarea Modelului

În această etapă, modelul a fost antrenat utilizând funcția fit() din TensorFlow, monitorizând performanța prin intermediul TensorBoard. Hiperparametrii, precum rata de învățare și numărul de epoci, au fost ajustați pentru a optimiza performanța. Setul de date a fost împărțit în seturi de antrenare și validare pentru a evalua capacitatea modelului de a generaliza pe date noi [9].

E. Evaluarea Modelului

Pentru a evalua acuratețea modelului, au fost utilizate metrici precum acuratețea și rata de eroare. Acuratețea măsoară proporția de predicții corecte realizate de model, în timp ce rata de eroare indică frecvența predicțiilor incorecte. Aceste metrici sunt esențiale pentru determinarea performanței generale a modelului în verificarea semnăturilor [25].

F. Testare

Modelul a fost testat pe un set de date rezervat pentru validare, permițând evaluarea performanței pe date neutilizate în timpul antrenării și oferind o estimare realistă a capacității de generalizare. Rezultatele obținute au demonstrat capacitatea modelului de a distinge eficient între semnăturile autentice și cele falsificate [26].

G. Generare Rezultate

După testare, modelul a fost utilizat pentru a verifica semnăturile din imagini, determinând autenticitatea acestora. Acest proces facilitează verificarea automată a semnăturilor, având aplicații practice în domenii precum securitatea documentelor și autentificarea identității. [27].

IV. FUNCȚIONAREA APLICAȚIEI DE VERIFICARE A SEMNĂTURILOR

Aplicația a fost implementată utilizând Python, iar funcționalitățile principale includ procesarea imaginilor, construirea algoritmului de clasificare și gestionarea bazei de date. Principalele biblioteci utilizate sunt Tkinter (pentru interfața grafică), OpenCV (pentru procesarea imaginilor) și SQLite (pentru gestionarea bazei de date).

Imaginile semnăturilor sunt încărcate din baza de date CEDAR, care conține semnături originale și falsificate. Fiecare imagine este supusă următoarelor etape de preprocesare:

Conversia în tonuri de gri: Imaginea este convertită pentru a reduce complexitatea.

Redimensionarea: Toate imaginile sunt redimensionate la 128x32 pixeli pentru a asigura uniformitatea în timpul antrenării și predicției.

Normalizarea: Valorile pixelilor sunt scalate între 0 și 1.

Codul de preprocesare poate fi văzut în Figura 3, unde funcția incarca_dataset_cedar() realizează toate aceste operații. Această funcție este esențială pentru standardizarea datelor de intrare înainte de utilizarea modelului.

```
imagine = cv2.imread(cale_imagine, cv2.IMREAD_GRAYSCALE)
if imagine is not None:
    imagine = cv2.resize(imagine, (128, 32))  # Redimensionare imagine
    imagine = imagine / 255.0  # Normalizare imagine
    imagini.append(imagine)
    etichete.append(eticheta)
```

Figura 3. Codul de preprocesare a imaginilor din data_loader.py.

Modelul utilizat pentru clasificarea semnăturilor este un CNN (Convolutional Neural Network) construit folosind biblioteca TensorFlow/Keras. Arhitectura sa este formată din:

Straturi Convoluționale: Extrage trăsături vizuale esențiale ale semnăturii, precum contururi și texturi.

Pooling: Reduce dimensiunile datelor, menținând informațiile relevante.

Straturi Dense: Clasifică imaginea ca "Originală" sau "Falsă" folosind funcția de activare sigmoid.

Codul modelului este ilustrat în Figura 4, unde funcția construieste_model() definește arhitectura CNN.

```
def construieste_model(dimensiune_intrare):
    """
    Construiește un model de rețea neuronală convoluțională pentru verificarea semnăturilor.
    """
    try:
        model = models.Sequential([
            layers.Conv2D(32, (3, 3), activation='relu', input_shape=dimensiune_intrare),
            layers.MaxPooling2D((2, 2)),
            layers.Conv2D(64, (3, 3), activation='relu'),
            layers.MaxPooling2D((2, 2)),
            layers.Flatten(),
            layers.Dense(128, activation='relu'),
            layers.Dense(1, activation='sigmoid')  # Clasificare binară: Originală (1) sau Falsă (0)
        ])
        logging.info("Modelul a fost construit cu succes.")  # Logare succes
        return model
    except Exception as e:
        logging.error(f"Eroare la construirea modelului: {e}")  # Logare eroare
        raise
```

Figura 4. Codul modelului CNN din model.py.

Aplicația utilizează SQLite pentru a stoca informațiile despre semnături. Acțiunile asupra bazei de date includ:
Adăugarea unei semnături: Imaginea selectată este salvată în baza de date folosind funcția adauga_semnatura() (Figura 5).

```
def adauga_semnatura(cale_db, cale_imagine):
    """
    Adaugă o semnătură în baza de date și returnează ID-ul acesteia.
    """
    if not os.path.exists(cale_imagine):
        logging.error(f"Fișierul {cale_imagine} nu există.")
        return None

    try:
        conn = sqlite3.connect(cale_db)
        cursor = conn.cursor()
        cursor.execute('SELECT id FROM semnaturi WHERE cale_imagine = ?', (cale_imagine,))
        if cursor.fetchone():
            logging.info(f"Imaginea {cale_imagine} există deja în baza de date.")
            return None

        cursor.execute('INSERT INTO semnaturi (cale_imagine) VALUES (?)', (cale_imagine,))
        conn.commit()
        return cursor.lastrowid
    except sqlite3.Error as e:
        logging.error(f"Eroare la adăugarea semnăturii: {e}")
    finally:
        if conn:
            conn.close()
```

Figura 5. Adăugarea semnăturii în baza de date din database.py.

Ștergerea unei semnături: O semnătură poate fi ștearsă pe baza ID-ului prin funcția sterge_semnatura() (Figura 6).

```
def sterge_semnatura(cale_db, id_semnatura):
    """
    Șterge o semnătură din baza de date după ID și reindexează ID-urile.
    """
    try:
        conn = sqlite3.connect(cale_db)
        cursor = conn.cursor()
        cursor.execute('DELETE FROM semnaturi WHERE id = ?', (id_semnatura,))
        conn.commit()
        reindexeaza_iduri(conn)
        return True
    except sqlite3.Error as e:
        logging.error(f"Eroare la ștergerea semnăturii: {e}")
        return False
    finally:
        if conn:
            conn.close()
```

Figura 6. Ştergerea unei semnături din baza de date.

Verificarea existenței: Înainte de adăugare, aplicația verifică dacă imaginea există deja.

Interfața grafică, realizată cu Tkinter, oferă o interacțiune intuitivă cu utilizatorul. Principalele butoane și funcțiile acestora sunt:

Încarcă Imagine: Deschide un dialog pentru selecția unei imagini.

Verifică Semnătura: Procesează imaginea încărcată și utilizează modelul CNN pentru clasificare.

Adaugă Semnătura: Adaugă imaginea în baza de date, dacă aceasta nu există deja.

Șterge Semnătura: Permite ștergerea unei semnături existente.

În Figura 7 este prezentată o parte din codul interfeței grafice a aplicației.

```
# Configurare logging
logging.basicConfig(level=logging.INFO, format='%(asctime)s - %(levelname)s - %(message)s')

class AplicatieSemnaturi:
    def __init__(self, root, model, db_path):
        self.root = root
        self.root.title("Sistem de Verificare a Semnăturilor")
        self.model = model
        self.db_path = db_path
        self.signature_path = None
        self.img_tk = None  # Inițializare atribut pentru a păstra referința imaginii

        # Componente GUI
        tk.Label(root, text="Sistem de Verificare a Semnăturilor", font=("Helvetica", 16)).pack(pady=10)
        self.image_label = tk.Label(root)
        self.image_label.pack(pady=10)

        culoare_buton = '#ADD8E6'  # Codul hexazecimal pentru albastru deschis

        tk.Button(root, text="Încarcă Imagine", command=self.incarca_imagine, bg=culoare_buton).pack(pady=5)
        tk.Button(root, text="Verifică Semnătura", command=self.verifica_semnatura, bg=culoare_buton).pack(pady=5)
        tk.Button(root, text="Adaugă Semnătura", command=self.adauga_semnatura, bg=culoare_buton).pack(pady=5)
        tk.Button(root, text="Șterge Semnătura", command=self.sterge_semnatura, bg=culoare_buton).pack(pady=5)
```

Figura 7. Codul interfeței grafice din gui.py.

A. *Funcționarea aplicației*

La rularea aplicației, utilizatorul interacționează cu interfața grafică pentru a încărca o imagine a semnăturii. După selectarea imaginii, aceasta este afișată în interfață, iar utilizatorul poate iniția procesul de verificare. Aplicația preprocesează imaginea și o compară cu semnăturile din baza de date, utilizând un model CNN antrenat pentru a distinge între semnături autentice și falsificate. Rezultatul este afișat sub forma unui mesaj care indică dacă semnătura este originală sau falsă, împreună cu un procent de predictie.

Figura 3 arată interfața principală a aplicației, cu cele patru butoane funcționale.

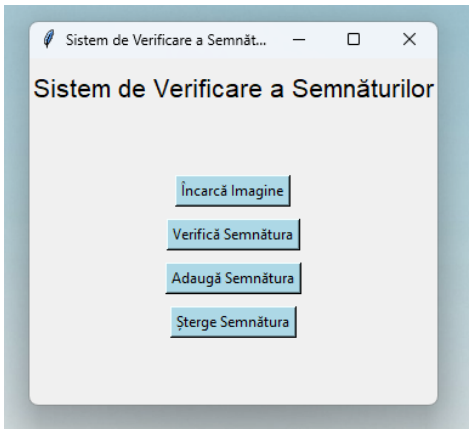

Figura 3. Interfața aplicației

B. *Scenarii de utilizare*

Utilizatorul apasă butonul "Încarcă Imagine" și selectează fișierul din sistemul local. Imaginea este afișată în interfață.

Figura 4 prezintă selectarea unei imagini din directorul full_org al bazei de date CEDAR.

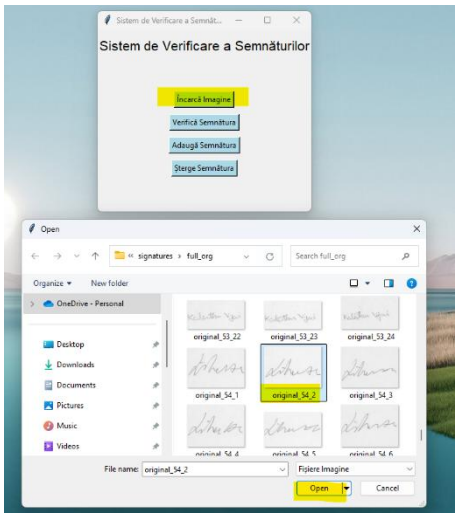

Figure 4 Butonul Încarcă Imagine.

După încărcarea imaginii, utilizatorul apasă butonul "Verifică Semnătura". Aplicația procesează imaginea și utilizează modelul CNN pentru a determina dacă semnătura este originală sau falsă. Rezultatul este afișat într-un mesaj care include probabilitatea de clasificare.

Figura 5 arată rezultatul verificării pentru o semnătură originală, cu o probabilitate de 88.84%.

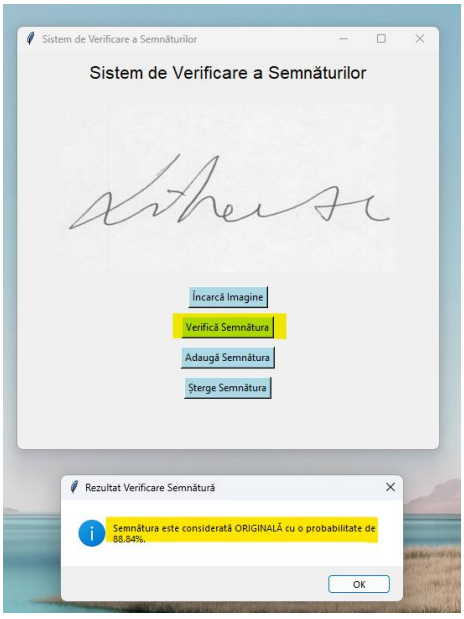

Figure 5 Verificarea semnăturii.

Dacă utilizatorul dorește să adauge semnătura în baza de date, acesta apasă butonul "Adaugă Semnătura". Aplicația verifică dacă imaginea există deja în baza de date și, în caz afirmativ, afișează un mesaj informativ. *Figura 6* prezintă mesajul care informează utilizatorul că semnătura există deja în baza de date.

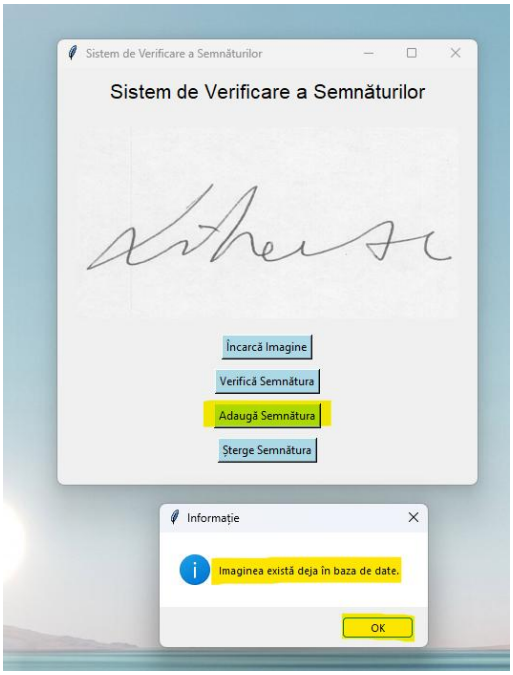

Figure 6. Adăugarea unei semnături

Utilizatorul poate vizualiza toate semnăturile existente împreună cu ID-urile lor, apoi introduce ID-ul semnăturii dorite pentru ștergere. Aplicația confirmă ștergerea printr-un mesaj de succes. *Figura 7* arată lista semnăturilor existente și ștergerea unei semnături cu ID-ul corespunzător.

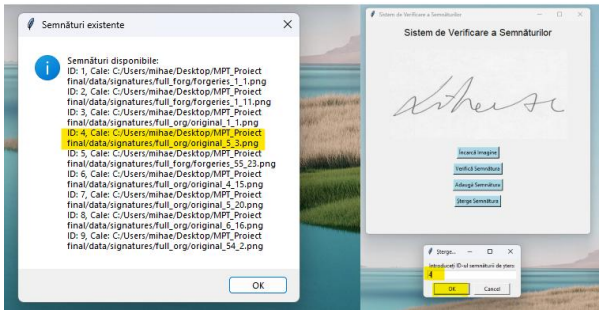

Figure 7. Ștergerea unei semnături

Utilizatorul încarcă o semnătură falsificată din directorul full_forg. După verificare, aplicația identifică semnătura ca falsă și afișează probabilitatea corespunzătoare. *Figura 8* prezintă rezultatul unei semnături falsificate, cu o probabilitate de 55.82%.

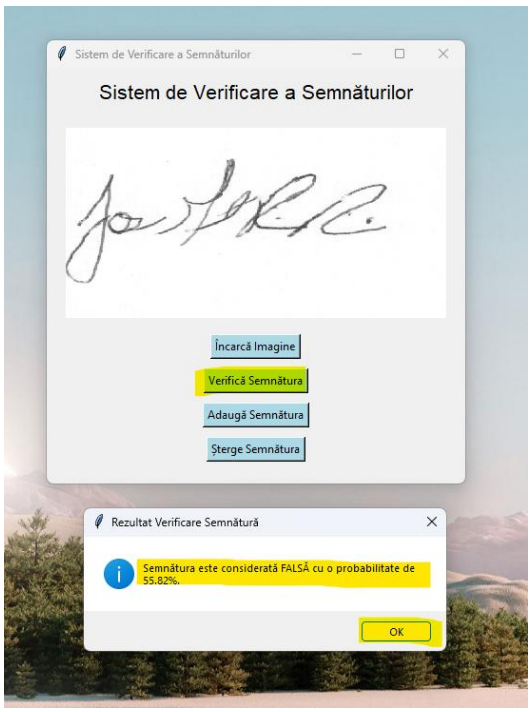

Figure 8. Verificarea unei semnături falsificate

Prin integrarea componentelor descrise, aplicația oferă un sistem eficient pentru verificarea semnăturilor, bazându-se pe modelul CNN antrenat și o interfață intuitivă pentru utilizatori.

V. CONCLUZII

Acest studiu a prezentat prezentat o metodă pentru construirea unui sistem de verificare a semnăturilor utilizând rețele neuronale convoluționale și baza de date CEDAR. Am detaliat etapele de preprocesare a datelor, definirea arhitecturii modelului, antrenarea și testarea pe un set de date de validare. Metodologia oferă o bază solidă pentru dezvoltarea unui sistem automat de verificare a semnăturilor, iar performanța modelului poate fi îmbunătățită prin ajustarea hiperparametrilor, utilizarea unor seturi de date suplimentare și aplicarea tehnicilor de augmentare a datelor. Acest framework constituie o bază promițătoare pentru optimizarea și extinderea aplicațiilor de verificare automată a semnăturilor.

BIBLIOGRAFIE

[1]. R. D. Rai și J. S. Lather, „Handwritten Signature Verification using TensorFlow”, în 2018 3rd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), Bangalore, India: IEEE, mai 2018, pp. 2012–2015. doi: 10.1109/RTEICT42901.2018.9012273.

[2]. S. Kulkarni, Y. Raut, și S. Shrivastava, „Recognition of Handwritten Digit Using CNN in Python with Tensorflow”, vol. 8, nr. 4, 2021.

[3]. A. Pradhan, „Forgery Detection on Handwritten Signatures using Convolutional Neural Networks”, *Int. J. Res. Appl. Sci. Eng. Technol.*, vol. 11, nr. 11, pp. 1693–1709, nov. 2023, doi: 10.22214/ijraset.2023.56879.

[4]. F. Ertam and G. Aydin, “Data classification with deep learning using Tensorflow,” 2017 International Conference on Computer Science and Engineering (UBMK), 2017.

[5]. „Data classification with deep learning using Tensorflow | IEEE Conference Publication | IEEE Xplore”. Data accesării: 10 noiembrie 2024. [Online]. Disponibil la: https://ieeexplore.ieee.org/document/8093521

[6]. S. Hochreiter și J. Schmidhuber, „Long Short-Term Memory”, *Neural Comput.*, vol. 9, nr. 8, pp. 1735–1780, nov. 1997, doi: 10.1162/neco.1997.9.8.1735.

[7]. A. Graves, S. Fernandez, F. Gomez, și J. Schmidhuber, „Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks”.

[8]. Z. Wang, M. Muhammat, N. Yadikar, A. Aysa, și K. Ubul, „Advances in Offline Handwritten Signature Recognition Research: A Review”, *IEEE Access*, vol. 11, pp. 120222–120236, 2023, doi: 10.1109/ACCESS.2023.3326471.

[9]. R. Smith, „An Overview of the Tesseract OCR Engine”, în *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, sep. 2007, pp. 629–633. doi: 10.1109/ICDAR.2007.4376991.

[10]. ShubhangiS. Rokade, S. K. Singh, S. Bansod, și P. Pal, „An Offline Signature Verification Using Deep Convolutional Neural Networks”, în *2023 Third International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies (ICAECT)*, Bhilai, India: IEEE, ian. 2023, pp. 1–4. doi: 10.1109/ICAECT57570.2023.10117669.

[11]. „CEDAR Handwriting Recognition”. Data accesării: 12 decembrie 2024. [Online]. Disponibil la: https://cedar.buffalo.edu/handwriting.html

[12]. U. Marti and H. Bunke. A full English sentence database for off-line handwriting recognition. In Proc. of the 5th Int. Conf. on Document Analysis and Recognition, pages 705 - 708, 1999.

[13]. U. Marti and H. Bunke. Handwritten Sentence Recognition. In Proc. of the 15th Int. Conf. on Pattern Recognition, Volume 3, pages 467 - 470, 2000.

[14]. M. Zimmermann and H. Bunke. Automatic Segmentation of the IAM Off-line Database for Handwritten English Text. In Proc. of the 16th Int. Conf. on Pattern Recognition, Volume 4, pages 35 - 39, 2000.

[15]. U. Marti and H. Bunke. The IAM-database: An English Sentence Database for Off-line Handwriting Recognition. Int. Journal on Document Analysis and Recognition, Volume 5, pages 39 - 46, 2002.

[16]. S. Johansson, G.N. Leech and H. Goodluck. Manual of Information to accompany the Lancaster-Oslo/Bergen Corpus of British English, for use with digital Computers. Department of English, University of Oslo, Norway, 1978.

[17]. W. G. Hatcher și W. Yu, „A Survey of Deep Learning: Platforms, Applications and Emerging Research Trends”, IEEE Access, vol. 6, pp. 24411–24432, 2018, doi: 10.1109/ACCESS.2018.2830661.

[18]. Y. Muhtar, W. Kang, A. Rexit, Mahpirat, și K. Ubul, „A Survey of Offline Handwritten Signature Verification Based on Deep Learning”, în 2022 3rd International Conference on Pattern Recognition and Machine Learning (PRML), Chengdu, China: IEEE, iul. 2022, pp. 391–397. doi: 10.1109/PRML56267.2022.9882188.

[19]. Y. Muhtar, W. Kang, A. Rexit, Mahpirat, și K. Ubul, „A Survey of Offline Handwritten Signature Verification Based on Deep Learning”, în 2022 3rd International Conference on Pattern Recognition and Machine Learning (PRML), Chengdu, China: IEEE, iul. 2022, pp. 391–397. doi: 10.1109/PRML56267.2022.9882188.

[20]. H.-M. Chang, S.-C. Lin, P. S. Chen, și Y.-H. Hung, „A verification protocol of mobile payment based on signature recognition”, în 2015 International Carnahan Conference on Security Technology (ICCST), Taipei, Taiwan: IEEE, sep. 2015, pp. 379–384. doi: 10.1109/CCST.2015.7389714.

[21]. Z. Wang, M. Muhammat, N. Yadikar, A. Aysa, și K. Ubul, „Advances in Offline Handwritten Signature Recognition Research: A Review”, IEEE Access, vol. 11, pp. 120222–120236, 2023, doi: 10.1109/ACCESS.2023.3326471.

[22]. Y. Wang, J. Zheng, și Y. Zhou, „An Efficient Offline Signature Verification Method Based on Improved Feature Extraction”, în 2022 2nd International Conference on Computer Science, Electronic Information Engineering and Intelligent Control Technology (CEI), Nanjing, China: IEEE, sep. 2022, pp. 609–612. doi: 10.1109/CEI57409.2022.9950165.

[23]. Shubhangi S. Rokade, S. K. Singh, S. Bansod, și P. Pal, „An Offline Signature Verification Using Deep Convolutional Neural Networks”, în 2023 Third International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies (ICAECT), Bhilai, India: IEEE, ian. 2023, pp. 1–4. doi: 10.1109/ICAECT57570.2023.10117669.

[24]. Zhili Chen, Xinghua Xia, și Fangjun Luan, „Automatic online signature verification based on dynamic function features”, în 2016 7th IEEE International Conference on Software Engineering and Service Science (ICSESS), Beijing, China: IEEE, aug. 2016, pp. 964–968. doi: 10.1109/ICSESS.2016.7883226.

[25]. P. Y. Simard, D. Steinkraus, și J. C. Platt, „Best practices for convolutional neural networks applied to visual document analysis”, în Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings., aug. 2003, pp. 958–963. doi: 10.1109/ICDAR.2003.1227801.

[26]. Md. S. Sultana, S. Geethanjali, și M. Ramesh, „Comparative Model of Offline Signature Verification System”, în 2023 5th International Conference on Smart Systems and Inventive Technology (ICSSIT), Tirunelveli, India: IEEE, ian. 2023, pp. 1573–1578. doi: 10.1109/ICSSIT55814.2023.10060946.