**UNIVERSITATEA DE STAT DIN MOLDOVA**
**ȘCOALA DOCTORALĂ ȘTIINȚE FIZICE, MATEMATICE, ALE INFORMAȚIEI ȘI INGINEREȘTI**

Cu titlu de manuscris
C.Z.U.: 004:[94(478):008]

**BUMBU TUDOR**

# TEHNOLOGII ȘI RESURSE INFORMAȚIONALE PENTRU DIGITIZAREA ȘI PROCESAREA TEXTELOR DIN PATRIMONIUL ISTORICO-CULTURAL

**121.03 – PROGRAMAREA CALCULATOARELOR**

**Teză de doctor în informatică**

**Autor:** Bumbu Tudor
**Conducător științific:** Cojocaru Svetlana, mem. cor., prof. cerc., dr. hab. în informatică.
**Comisia de îndrumare** Gaindric Constantin, mem. cor., prof. cerc., dr. hab. în informatică;
Țițchiev Inga, dr. în informatică, conf. univ.;
Burțeva Liudmila, dr. în informatică, conf. cerc.

**CHIȘINĂU, 2023**

# CUPRINS


**ADNOTARE** 4

**ANNOTATION** 5

**АННОТАЦИЯ** 6

**INTRODUCERE** 7

**1. INSTRUMENTE ȘI METODE DE PROCESAR A DOCUMENTELOR ISTORICE** **14**

**1.1. Definiții și noțiuni de bază** 14

**1.2. Metode, instrumente, resurse și platforme de digitizare a documentelor istorice** 15

**1.3. Documente istorice românești tipărite cu alfabete chirilice** 43

**1.4. Concluzii la capitolul 1** 45

**2. TEHNOLOGII DE PROCESARE A TEXTELOR ROMÂNEȘTI DIN SEC. XVII-XX** **47**

**2.1. Descrierea procesului de recunoaștere optică a caracterelor** 47

**2.2. Preprocesarea imaginilor vechi** 49

**2.3. Crearea limbii utilizatorului și adăugarea dicționarului** 53

**2.4. Descrierea procesului de instruire cu FR12, crearea șabloanelor** 55

**2.5. Modele OCR aplicate pe texte tipărite în secolul XVII** 57

**2.6. Evaluarea OCR a documentelor din secolul XVII** 60

**2.7. Clasificarea fonturilor din secolul XVII** 64

**2.8. Transliterarea din alfabetul chirilic român în alfabetul modern** 80

**2.9. Instrumentul software de transliterare din ACR în AMR** 84

**2.10. Instrumente de aliniere a textelor vechi la cele moderne** 86

**2.11. Concluzii la capitolul II** 93

**3. PLATFORMĂ DE DIGITIZARE** **95**

**3.1. Arhitectura platformei de digitizare** 96

**3.2. Module de recunoaștere optică a caracterelor** 102

**3.3. Module de transliterare a textelor** 105

**3.4. Module de gestionare a documentului digitizat** 107

**3.5. Aplicație de digitizare** 109

**3.6. Concluzii la capitolul 3** 118

**CONCLUZII GENERALE ȘI RECOMANDĂRI** 119

**BIBLIOGRAFIE** 122

**DECLARAŢIA PRIVIND ASUMAREA RĂSPUNDERII** 134

**CURRICULUM VITAE** 135

# ADNOTARE

**Bumbu Tudor: "Tehnologii și resurse informaționale pentru digitizarea și procesarea textelor din patrimoniul istorico-cultural".**

**Teză de doctor în informatică, Chișinău, 2023.**

**Structura tezei:** teza este scrisă în limba română și constă din introducere, 3 capitole, concluzii generale și recomandări, bibliografie din 140 de titluri. Teza conține 120 de pagini cu text de bază, 59 figuri și 10 tabele. Rezultatele obținute sunt publicate în 17 lucrări științifice.

**Cuvinte-cheie:** digitizare, patrimoniul de limbă română, documente chirilice, rețele neurale, modele OCR, transliterare, platformă de digitizare.

**Scopul lucrării:** elaborarea instrumentelor informatice pentru procesarea patrimoniului de limbă română tipărit în secolele 17-20.

**Obiectivele cercetării:** crearea unei colecții de resurse scanate pentru antrenarea modelelor OCR și elaborarea dicționarelor OCR; elaborarea unei tehnologii OCR pentru documentele românești tipărite în secolele 17-20; dezvoltarea algoritmilor de transliterare din grafie chirilică în cea latină pentru o varietate de alfabete; dezvoltarea unei platforme de digitizare pentru procesarea documentelor chirilice românești.

**Noutatea și originalitatea științifică:** constau în cercetarea și elaborarea tehnologiei pentru soluționarea problemei de recunoaștere și transliterare a documentelor chirilice românești tipărite în secolele 17-20.

**Rezultatul obținut care contribuie la soluționarea unei probleme științifice importante** îl constituie dezvoltarea tehnologiei de recunoaștere optică a caracterelor și transliterare din grafia chirilică în cea latină a documentelor chirilice românești tipărite în secolele 17-20, în condițiile existenței unei varietăți mari de alfabete și fonturi.

**Semnificația teoretică a lucrării:** este determinată de obținerea unei tehnologii care permite conversia documentelor românești din alfabetul chirilic în cel latin, cu aplicarea și dezvoltarea metodelor bazate pe rețele neurale.

**Valoarea aplicativă a lucrării:** constă în elaborarea unei platforme de digitizare, care aduce un aport substanțial la automatizarea reeditării documentelor vechi, fiind un instrument util pentru un cerc larg de utilizatori.

**Implementarea rezultatelor lucrării:** Instrumentele de digitizare au fost utilizate pentru recunoașterea parțială sau completă a unor cărți românești tipărite în alfabetul chirilic. Instrumentarul de recunoaștere și transliterare a fost instalat pentru utilizare în cadrul Bibliotecii Academiei Române și a Bibliotecii Științifice „Andrei Lupan".

# ANNOTATION

**Bumbu Tudor: "Technologies and Information Resources for the Digitization and Processing of Texts from the Cultural Heritage."**

**Doctoral thesis in Computer Science, Chisinau, 2023.**

**The structure of the thesis:** the thesis is written in Romanian and consists of an introduction, 3 chapters, general conclusions and recommendations, a bibliography of 140 titles. The thesis contains 120 pages of basic text, 59 figures, and 10 tables. The obtained results are published in 17 scientific papers.

**Keywords:** digitization, Romanian language heritage, Cyrillic documents, neural networks, OCR models, transliteration, digitization platform.

**The aim of the paper:** development of computer tools for processing printed Romanian language heritage of $17^{th}$-$20^{th}$ centuries.

**Research objectives:** Creation of a collection of scanned resources for training OCR models and developing OCR dictionaries; Development of OCR technology for Romanian printed documents from the 17th to 20th centuries; Development of transliteration algorithms from Cyrillic to Latin script for a variety of alphabets; Development of a digitization platform for processing Romanian Cyrillic documents.

**The novelty and scientific originality:** are based on the research and development of technology for solving the problem of recognition and transliteration of Romanian Cyrillic documents printed in the 17th to 20th centuries.

**The result obtained that contributes to solving an important scientific problem:** is the development of optical character recognition technology and transliteration from Cyrillic to Latin script of Romanian Cyrillic documents printed in the 17th to 20th centuries, under the conditions of a wide variety of alphabets and fonts.

**Theoretical significance of the paper:** is determined by obtaining a technology that allows the conversion of Romanian documents from Cyrillic alphabet to Latin, with the application and development of methods based on neural networks.

**The practical value of the paper:** is based on the development of a digitization platform, which brings a substantial contribution to the automation of republishing old documents, being a useful tool for a wide range of users.

**Implementation of the paper results:** Digitization tools have been used for partial or complete recognition of some Romanian books printed in Cyrillic alphabet. The recognition and transliteration tools have been installed for use at the Romanian Academy Library and the "Andrei Lupan" Scientific Library.

# АННОТАЦИЯ

**Тудор Бумбу: “Технологии и информационные ресурсы для оцифровки и обработки текстов из культурного наследия”**

**Докторская диссертация по информатике, Кишинёв, 2023 год.**

**Структура диссертации:** диссертация написана на румынском языке и состоит из введения, 3 глав, общих выводов и рекомендаций, библиографии из 140 наименований. Диссертация содержит 120 страниц основного текста, 59 иллюстраций и 10 таблиц. Полученные результаты опубликованы в 17 научных работах.

**Ключевые слова:** оцифровка, наследие румынского языка, кириллические документы, нейронные сети, модели OCR, транслитерация, платформа для оцифровки.

**Цель работы:** разработка компьютерных инструментов для обработки печатного наследия румынского языка XVII-XX веков.

**Задачи исследования:** создание коллекции сканированных ресурсов для обучения моделей OCR и разработка словарей OCR; разработка технологии OCR для румынских печатных документов в XVII-XX веках; разработка алгоритмов транслитерации с кириллицы на латиницу; разработка платформы оцифровки для обработки румынских кириллических документов.

**Научная новизна и оригинальность работы:** основываются на исследовании и разработке технологии для решения проблемы распознавания и транслитерации румынских кириллических документов, напечатанных в XVII-XX веках на различных алфавитах.

**Полученный результат, который способствует решению важной научной проблемы:** разработка технологии оптического распознавания символов и транслитерации с кириллицы на латиницу румынских кириллических документов, напечатанных в XVII-XX веках, в условиях большого разнообразия алфавитов и шрифтов.

**Теоретическая значимость работы:** определяется получением технологии, которая позволяет конвертировать румынские документы из кириллического алфавита в латинский, с применением и разработкой методов, основанных на нейронных сетях.

**Прикладная ценность работы:** заключается в разработке платформы для оцифровки, которая вносит существенный вклад в автоматизацию переиздания старых документов, являясь полезным инструментом для широкого круга пользователей.

**Реализация результатов работы:** Инструменты оцифровки были использованы для частичного или полного распознавания ряда румынских книг, напечатанных кириллицей, инструменты распознавания и транслитерации были установлены для использования в Библиотеке Румынской Академии и научной библиотеке “Andrei Lupan”.

# INTRODUCERE

**Actualitatea și importanța temei de cercetare.** Digitizarea ocupă un loc de frunte în tehnologiile secolului XXI. Încă în anul 2011, Comisia Europeană a venit cu un document de recomandări privind digitizarea și accesibilitatea online a materialului cultural și conservarea digitală [1], în care menționa că „dezvoltarea procesului de digitizare a materialului aflat în biblioteci, arhive și muzee ar trebui să fie încurajată în continuare, pentru a se garanta faptul că Europa își menține poziția de actor principal pe plan internațional în domeniul culturii și al conținutului creativ și că își utilizează bogăția materialului cultural, în cel mai bun mod cu putință", îndemnând statele membre să își intensifice investițiile în acest domeniu.

Recomandarea a fost inclusă drept acțiune de politici în mai multe țări (nu doar cele din UE), dezvoltându-se o întreagă industrie ce oferă servicii de scanare, recunoaștere și alte activități adiacente, problema digitizării și conservării tezaurului cultural reprezentând un domeniu prioritar din agenda digitală pentru Europa [2].

Digitizarea pe scară largă, unde acțiunile sunt scanarea și stocarea imaginilor, a început odată cu proiectul Gutenberg [3], inițiat în anii '70, iar mai târziu a apărut biblioteca digitală Hathi Trust [4], colecția de un milion de cărți [5] și proiectul de digitizare Google Books [6]. Cu toate că aceste proiecte soluționează problema conservării patrimoniului tipărit, digitizarea prin scanare a materialelor tipărite poate fi considerată doar punctul de plecare în ceea ce privește conservarea cunoștințelor incluse în ele și facilitarea accesului către acestea.

În pofida faptului că mai multe documente pot fi găsite și citite online, ele nu pot fi procesate automat, deoarece în cele mai dese cazuri sunt expuse doar în format de imagine, și nu cel de text lizibil (sau editabil) automat, de către mașină. Prin urmare, provocarea automatizării procesului de transformare a documentelor în text lizibil pentru calculator, deci editabil, revine aplicațiilor de învățare automată și viziune computerizată, și anume celor de recunoaștere optică a caracterelor (OCR – Optical Character Recognition). Vom demonstra în această lucrare că sarcina respectivă nu poate fi întotdeauna considerată drept una trivială, deoarece diapazonul de variație a materialului-sursă (calitatea documentului și volumul lui, perioada editării, condițiile de păstrare etc.) este extrem de mare. Cu toate acestea, punerea la dispoziție a documentelor de patrimoniu cultural digitizat sub formă editabilă a fost și este considerată în continuare o necesitate. Acest lucru este subliniat în mod deosebit în raportul de referință [7], care avertizează că Europa se află în pericol de a intra într-o nouă eră întunecată dacă nu se creează mijloace suficiente de conservare și facilitare a accesului la materialul de patrimoniu cultural. Ca urmare, au fost finanțate mai multe proiecte la scară largă care se ocupă de OCR-ul tipăriturilor istorice în contextul digitizării în masă,

cel mai important fiind proiectul IMPACT [8, 9] care presupune îmbunătățirea accesului la text și proiectul OCR pentru tipăriturile moderne timpurii – eMOP[1] [9].

Abordarea acestei chestiuni pentru tipăriturile românești se confruntă cu provocări distincte: evoluția complexă a limbii noastre de-a lungul timpului, disponibilitatea limitată a resurselor, care, cu regret, nu sunt păstrate într-un mod concentrat, precum și o diversitate semnificativă de alfabete folosite la editarea acestor resurse. Obstacolele întâmpinate la digitizarea și conservarea acestui tezaur țin de recunoașterea corectă a literelor chirilice, dar și de inexistența unui lexicon adecvat perioadei de tipărire a resurselor vechi [10]. În particular, problema creării resurselor lingvistice, digitizarea și procesarea textelor ce fac parte din patrimoniul cultural din diverse perioade istorice este actuală pentru mai multe țări europene [11-16].

Prin Hotărârea Guvernului Republicii Moldova nr. 857 din 31 octombrie 2013 a fost aprobată Strategia națională de dezvoltare a societății informaționale „Moldova Digitală 2020", precum și planul de acțiuni pentru punerea în aplicare a acesteia. Acest act normativ, în pofida faptului că prevederile lui nu au fost realizate integral, a impulsionat activitățile de digitizare a colecțiilor de documente din bibliotecile și arhivele țării. Cu toate acestea, rămâne actuală soluționarea problemei digitizării patrimoniului cultural tipărit al Republicii Moldova, care ar oferi instrumente informatice capabile să proceseze documente din diferite perioade istorice, cu diferite alfabete, cu diverse vocabulare, păstrate în diverse condiții etc. – instrumente inteligente, de care ar putea beneficia atât cercetătorii, cât și publicul larg, oferindu-li-se posibilitatea operării cu uriașe colecții de date indexate.

**Scopul și obiectivele cercetării.** Scopul cercetării constă în fundamentarea și elaborarea instrumentelor informatice pentru procesarea patrimoniului de limbă română tipărit în secolele 17-20. Scopul propus a determinat necesitatea formulării următoarelor obiective:

- analiza și determinarea metodelor principale de preprocesare a documentelor vechi;
- crearea unei colecții de resurse scanate pentru antrenarea modelelor OCR și elaborarea dicționarelor OCR;
- elaborarea unei tehnologii OCR a documentelor românești tipărite în secolele 17-20;
- dezvoltarea algoritmilor de transliterare din grafie chirilică în cea latină;
- cercetarea și elaborarea metodelor de aliniere a textelor vechi vocabularului contemporan, elaborarea unui suport pentru aliniere;
- dezvoltarea unei platforme de digitizare pentru procesarea documentelor chirilice românești.

[1] https://emop.tamu.edu/

Realizarea obiectivelor propuse a contribuit la obținerea unor rezultate aplicative importante, încorporate în platforma de digitizare, utilizarea căreia facilitează accesul la patrimoniul cultural românesc tipărit în grafie chirilică.

**Noutatea și originalitatea științifică a lucrării** constau în cercetarea și elaborarea tehnologiei pentru soluționarea problemei de recunoaștere și transliterare a documentelor chirilice românești tipărite în secolele 17-20. Acest fapt permite procesarea eficientă și rapidă a documentelor menționate. Gradul de noutate și originalitate este reprezentat de:

- elaborarea tehnologiei OCR a documentelor românești tipărite în secolele 17-20;
- dezvoltarea algoritmilor de transliterare din alfabetul chirilic românesc în alfabetul modern (latin) românesc;
- elaborarea unei metode de clasificare a fonturilor utilizate la tipărirea textelor vechi;
- elaborarea unei metode de aliniere a textelor vechi la vocabularul modern utilizând tehnici de similaritate ale secvențelor.
- dezvoltarea unei platforme web de digitizare pentru procesarea documentelor chirilice.

**Problema științifică importantă rezolvată în domeniul de cercetare** este dezvoltarea tehnologiei de recunoaștere optică a caracterelor și transliterare din grafia chirilică în cea latină a documentelor chirilice românești tipărite în secolele 17-20, în condițiile existenței unei varietăți mari de alfabete și fonturi.

**Valoarea teoretică a lucrării** este determinată de obținerea unei tehnologii care permite conversia documentelor românești din alfabetul chirilic în cel latin, cu aplicarea și dezvoltarea metodelor bazate pe rețele neurale.

**Valoarea aplicativă a lucrării** constă în elaborarea unei platforme de digitizare, care aduce un aport substanțial la automatizarea reeditării documentelor vechi, fiind un instrument util pentru biblioteci și arhive în crearea conținutului digital, pentru cercetătorii din domeniul istoriei, filologiei, etc., dar și pentru un cerc larg de utilizatori, oferindu-le asistență la etapele de preprocesare, recunoaștere, și postprocesare a documentelor.

**Rezultatele științifice principale înaintate spre susținere:**

- S-a realizat o analiză comparativă a metodelor și instrumentelor folosite în preprocesarea imaginilor din documente vechi tipărite;
- S-a efectuat o analiză a platformelor de digitizare a documentelor vechi din patrimoniul cultural, dezvoltate în cadrul proiectelor europene;

- S-au colectat peste 900 de pagini din documente chirilice românești pentru a crea seturi de date și dicționare de cuvinte utilizate în procesul OCR;
- S-au antrenat peste 20 de modele OCR pentru recunoașterea documentelor chirilice românești tipărite între secolele 17-20;
- S-au dezvoltat algoritmi pentru transliterarea textelor chirilice din secolele 17-20 în alfabetul latin;
- S-a elaborat o metodă de clasificare a fonturilor bazată pe rețele neurale;
- S-a elaborat un sistem de aliniere a textelor vechi la vocabularul contemporan;
- S-a creat o platformă de digitizare pentru documentele chirilice românești, care include instrumente de preprocesare a imaginilor, modele OCR, aplicații pentru transliterarea din grafie chirilică în cea latină și module de editare a textelor recunoscute/transliterate.

**Implementarea rezultatelor lucrării.** Instrumentele de digitizare au fost utilizate pentru recunoașterea parțială sau completă a unor cărți românești tipărite în alfabetul chirilic, printre care se numără: *Noul Testament*, 1648; *De Obște Geografie*, 1795; *Ducere de mână către aritmetică*, 1785; *Fiziognomie*, 1785; *Legiuire*, 1818; *Epistolariul românesc*, 1841; *Elemente de aritmetică* de G. Asachi, 1836 și altele. Instrumentarul de recunoaștere și transliterare a fost instalat pentru utilizare în cadrul Bibliotecii Academiei Române și a Bibliotecii Științifice „Andrei Lupan". În ultimul caz sistemul a fost aplicat pentru digitizarea și transliterarea a 7 volume (din cele opt editate în perioada 1970-1981) ale "Enciclopediei Sovietice Moldovenești".

Platforma web de digitizare a fost plasată pe internet, la adresa https://digitizare.math.md/. În plus, unele modele OCR și module de transliterare pentru documente chirilice din secolul 20 au fost implementate în portalul Moldova Digitală[2], pe site-ul https://digi.emoldova.org/, pe acest portal fiind procesate peste 120 de articole. Autorul tezei a răspuns la mai multe solicitări în vederea recunoașterii și transliterării textelor tipărite în limba română cu caractere chirilice. Un astfel de exemplu îl constituie procesarea romanului "Labirintul" de Ariadna Șalari, pregătit pentru reeditare de către asociația "Arbor" (București) în cadrul proiectului cultural „Romanul românesc din stânga Prutului".

Totalul rezultatelor științifice obținute împreună cu aplicațiile dezvoltate au contribuit la crearea unui mediu în mare parte automatizat de digitizare a documentelor vechi chirilice românești, facilitând accesul la aceste resurse culturale și istorice.

[2] https://emoldova.org/

**Metodologia cercetării.** Pe parcursul cercetărilor efectuate în cadrul tezei au fost folosite metode din prelucrare a limbajului natural și învățare automată. Procesul de cercetare a fost unul complet, urmărindu-se parcurgerea riguroasă a fiecărei etape: definirea problemei, faza de documentare, modelarea problemei și emiterea ipotezelor de lucru, faza de testare, analiza rezultatelor și diseminarea lor.

**Aprobarea rezultatelor cercetării.** Rezultatele științifice obținute de autor în această teză au fost prezentate la conferințe științifice naționale și internaționale și au fost publicate în reviste internaționale.

**a) Principalele articole publicate în reviste științifice:**

- BUMBU, T. *Towards a Font Classification Model for Romanian Cyrillic Documents.* In: Computer Science Journal of Moldova. 2021, nr. 3(87), pp. 291-298. ISSN 1561-4042 [30];
- BUMBU, T. *On Alignment of Textual Elements in a Parallel Diachronic Corpus.* In: Computer Science Journal of Moldova. 2020, nr. 3(84), pp. 241-248. ISSN 1561-4042 [129];
- BUMBU, T., CAFTANATOV, O., MALAHOV, L. *Revitalization of the RM Folkloric Texts from the Second Half of the 20th Century and their Diachronic Analysis*. ROMAI J., v.14, no.2 (2018), pp. 33–40 [137];
- COJOCARU, S., COLESNICOV, A., MALAHOV, L., BUMBU, T., UNGUR, Ș. *On Digitization of Romanian Cyrillic Printings of the 17th-18th Centuries.* CSJM, vol.25, no.2 (74), 2017, pp.217-225 [133];
- COJOCARU, S., COLESNICOV, A., MALAHOV, L., BUMBU, T. *Optical Character Recognition Applied to Romanian Printed Texts of the 18th–20th Century.* CSJM, vol.24, no.1 (70), 2016 [81].

**b) Principalele lucrări prezentate la conferințe naționale și internaționale:**

- *Development of a platform for heterogeneous document recognition using convergent technology*. Workshop on Intelligent Information Systems WIIS 2022, October 06-08, 2022, Chisinau, Republic of Moldova [135];
- *Platform for Digitization of Heterogeneous Documents*. The 29th Conference on Applied and Industrial Mathematics CAIM 2022, August 25-27, 2022, Chisinau, Republic of Moldova [134];

- *Punctilog Compared to Dependency Grammar and Constituency Grammar*. Symposium on Logic and Artificial Intelligence SLAI2022, January 12-16, 2022, Louisiana, USA [131];
- *User Interface to Access Old Romanian Documents.* The 4th Conference of Mathematical Society of Moldova CMSM4'2017, June 25-July 2, 2017 [29];
- *Generation of the Romanian Cyrillic lexicon for the period 1967 – 1989.* The Fifth Conference of Mathematical Society of the Republic of Moldova. 28 septembrie - 1 octombrie 2019, Chișinău. Chișinău, Republica Moldova [138];
- *On Classification of 17th Century Fonts using Neural Networks*. Workshop on Intelligent Information Systems (WIIS2021), October 14-15, 2021, Chisinau, Republic of Moldova [119];
- *Discover the Moldovan Cultural Heritage through e-Moldova Portal by Using Crowdsourcing Concept*. Workshop on Intelligent Information Systems WIIS2021, October 14-15, 2021, Chisinau, Republic of Moldova [140];
- *Evaluarea Corpusului Diacronic Paralel cu Texte Românești din Noul Testament din 1648 & 1990*. Conferința științifică a doctoranzilor „Tendințe contemporane ale dezvoltării științei: viziuni ale tinerilor cercetători", ediția a 9-a, vol., 10 iunie 2020, Chișinău [128];
- *Building a Diachronic Parallel Corpus for the Alignment of the Old Romanian Texts.* Conference on Mathematical Foundations of Informatics MFOI-2019, July 3-6, 2019, Iasi, Romania [127];

Au fost publicate 17 lucrări științifice, cuprinzând 5 articole în reviste științifice (2 articole fără co-autori) și 12 articole în materialele conferințelor.

**Volumul și structura tezei.** Teza este scrisă în limba română, tehnoredactată la calculator, cu titlu de manuscris. Lucrarea are următoarea structură: introducere, 3 capitole, concluzii generale și recomandări și bibliografie.

**Sumarul compartimentelor tezei.** În compartimentul „Introducere" se evidențiază relevanța și importanța temei de cercetare, prezentând informații concise și actualizate despre stadiul recent al digitizării patrimoniului istorico-cultural. Sunt descrise scopul și obiectivele tezei, noutatea științifică a rezultatelor obținute, precum și valoarea teoretică și aplicativă a tezei, acestea fiind însoțite de demonstrarea și validarea rezultatelor.

**Capitolul 1** „*Instrumente și metode de procesare a documentelor istorice*" include o analiză a studiilor științifice referitoare la metodele, instrumentele și resursele pentru digitizarea

documentelor din patrimoniul istorico-cultural. Capitolul începe cu o prezentare a definițiilor și conceptelor legate de digitizare și recunoașterea documentelor vechi. Ulterior sunt descrise metodele și instrumentele existente pentru recunoașterea documentelor vechi, analiza fiind efectuată atât la nivel internațional, cât și realizat un studiu de caz pentru textele în limba română. Se descriu metodele fundamentale de preprocesare a imaginilor vechi și de postprocesare a textelor după recunoaștere. În plus, acest capitol analizează variantele de documente istorice românești tipărite cu alfabetul chirilic.

**Capitolul 2** *„Tehnologii de procesare a documentelor românești din sec. XVII-XX*" fundamentează abordările noastre în proiectarea tehnologiei de procesare a textelor istorice (tipărite în limba română cu caractere chirilice, începând cu secolul al XVII-lea), descriind metodele elaborate și argumentând utilizarea anumitor module din categoria celor existente. Acestea includ: module de preprocesare a imaginilor; modele OCR; modele de rețele neurale pentru clasificarea fonturilor; tehnologia de transliterare din alfabetul chirilic românesc în cel modern; suport pentru alinierea textelor vechi la cele moderne.

**Capitolul 3** *„Platformă pentru digitizarea documentelor chirilice românești"* este dedicat proiectării și descrierii platformei web care include instrumentarul de digitizare dezvoltat în această lucrare. Sunt prezentate modulele implementate în platformă, necesare pentru a realiza trei sarcini principale referitoare la digitizarea documentelor vechi românești: preprocesarea imaginii, recunoașterea optică și transliterarea textului recunoscut din grafie chirilică în latină. Procesul de digitizare cuprinde 7 etape de procesare a documentului, iar durata de timp necesară pentru digitizarea completă a unui document tipărit cu alfabetul chirilic românesc variază între 2 și 15 minute, în funcție de volumul documentului.

# 1. INSTRUMENTE ȘI METODE DE PROCESARE A DOCUMENTELOR ISTORICE

## 1.1. Definiții și noțiuni de bază

În acest compartiment vom defini noțiunile de bază care ne vor servi în expunerea ulterioară.

**Patrimoniul** este caracterizat în documentele UNESCO drept „moștenirea noastră din trecut, ceea ce trăim astăzi și ceea ce transmitem generațiilor viitoare” [17]. Se aliniază acestei definiții și lucrarea [18], unde se afirmă că „Patrimoniul cultural include locurile, lucrurile și practicile pe care o societate le consideră vechi, importante și demne de a fi conservate.” Același document UNESCO [17] definește și **patrimoniul digital,** specificând că acesta este constituit din „materiale computerizate, cu o valoare de durată, care ar trebui păstrate pentru generațiile viitoare.”

**Digitizarea cărților și a documentelor vechi** reprezintă una din modalitățile sigure de protejare și conservare a obiectelor de patrimoniu cultural, care totodată facilitează și accesul larg la utilizarea acestora.

**Recunoașterea optică a caracterelor sau *recunoașterea documentelor***, adesea abreviată ca OCR, este procesul de traducere mecanică sau electronică a imaginilor scanate ale textului tipărit, dactilografiat sau scris de mână în text editabil. Este utilizat pe scară largă pentru a converti cărți și documente în fișiere electronice, având un spectru larg de aplicații. OCR face posibilă editarea textului, căutarea unui cuvânt sau expresie, stocarea acestuia într-un mod mai compact, afișarea sau tipărirea unei copii și aplicarea tehnicilor precum traducerea automată, text-to-speech și text mining. OCR reprezintă și un domeniu de cercetare în recunoașterea modelelor, inteligența artificială și viziunea computerizată [19]. Sistemele informatice de OCR mimează capacitatea umană a creierului de a recunoaște literele, numerele și simbolurile dintr-o imagine. Sistemele OCR joacă un rol enorm în software-ul de recunoaștere a textului pe care mulți dintre noi se bazează astăzi utilizând Adobe Acrobat, Google Drive și alte sisteme similare. Primul motor OCR omni-font, capabil să recunoască textul imprimat în aproape orice font a fost dezvoltat de informaticianul și inventatorul american Ray Kurzweil [20].

**Adevăr de bază** (eng. *ground truth*) este informația despre care se știe că este reală sau adevărată, furnizată prin observare și măsurare directă (prin dovezi empirice), spre deosebire de informațiile furnizate prin inferență [21]. „Adevărul de bază” poate fi văzut ca un termen conceptual legat de cunoașterea adevărului cu privire la o anumită întrebare. Este rezultatul ideal

așteptat [22]. Clasele sau etichetele din exemplele de antrenare folosite la instruirea algoritmilor de învățare automată supervizată reprezintă *adevăruri de bază.*

**Acuratețea OCR la nivel de caracter** a unui document este calculată folosind formula:

$$A_{ch} = 1 - \frac{e}{ch} \quad (1)$$

unde *e* este numărul de caractere clasificate (recunoscute) greșit, iar *ch* este numărul total de caractere din document.

**Acuratețea OCR la nivel de cuvânt** a unui document este descrisă de formula:

$$A_{cuv} = 1 - \frac{e}{cuv} \quad (2)$$

unde *e* este numărul de cuvinte care conțin cel puțin o singură litera greșită, iar *cuv* este numărul total de cuvinte din document.

În continuare, unele noțiuni și definiții vor fi incluse direct în text sau sub formă de note de subsol.

### 1.2. Metode, instrumente, resurse și platforme de digitizare a documentelor istorice

Pentru a procesa documentele istorice este nevoie de metode și instrumente speciale. Acestea, de obicei, includ metode de procesare a imaginii, instrumente de recunoaștere optică a caracterelor, instrumente de procesare ulterioară sau postprocesare a textelor recunoscute, dar și metode de păstrare a rezultatelor obținute după procesare.

#### Descrierea comparativă a metodelor și instrumentelor de recunoaștere a documentelor istorice

Un număr semnificativ de documente istorice sunt deja scanate și stocate în baze de date de arhivă și portaluri. O sarcină importantă aici constă în a face astfel de documente ușor de accesat pentru a putea găsi informații și de a extrage cunoștințe. În primul rând, imaginile documentului trebuie convertite în text utilizând recunoașterea optică a caracterelor. Au fost propuse mai multe metode și instrumente OCR pentru procesarea documentelor istorice, totuși, unele excelează în anumite sarcini, în majoritatea cazurilor fiind produse comerciale, iar altele - gratuite – chiar dacă

oferă o funcționalitate relativ redusă în ceea ce privește acuratețea de recunoaștere a caracterelor, prezintă totuși instrumente potrivite pentru cercetare și experimentare.

Recunoașterea optică a caracterelor pentru documentele moderne tipărite folosind alfabetul latin funcționează foarte bine (acurateţe de peste 99% în majoritatea cazurilor) și este adesea considerată o problemă rezolvată [23]. Metodele OCR tradiționale disponibile în produsele software comerciale și gratuite funcționează după cum urmează: în timpul procesului OCR, o imagine a unei pagini tipărite este segmentată în caractere care sunt apoi comparate cu seturi de caracteristici abstracte ce descriu exemple de caractere învățate anterior dintr-un set de caractere ale unui font. Asemănarea dintre caracterele învățate și recunoscute, separarea clară a caracterelor negre de fundalul alb și ortografia modernă a cuvintelor contribuie la rezultate excelente de recunoaștere.

Până în 2014 documentele istorice reprezentau o limită severă pentru eficacitatea OCR, deoarece majoritatea motoarelor OCR disponibile au fost instruite pe larg cu fonturi moderne, dar din moment ce fonturile istorice sunt foarte diferite de cele moderne, ele necesită instruire separată de către echipe de specialiști. Pentru textele relativ vechi (secolul al XIX-lea), rezultatele OCR obținute de la motoarele comerciale sunt adesea mai puțin satisfăcătoare [24-25]. Chiar și fonturile *Antiqua* (forme de glife romane) ale tipăriturilor istorice duc adesea la o acurateţe de recunoaștere a caracterelor de aproximativ 85%, fiind recunoscute cu ABBYY FineReader, un produs comercial de top [26]. Începând cu 2019, compania ABBYY a implementat noi algoritmi de inteligență artificială [27], care contribuie substanțial la recunoașterea limbilor bazate pe un script complex, cum ar fi chineza, japoneza și coreeana. Prin urmare, se folosesc modele diferite pentru scrisul latin, chirilic, arab și altele. De exemplu, pentru scrierea arabă se folosește o abordare cap-coadă pentru recunoașterea cuvintelor fără separarea caracterelor. O arhitectură specială, combinată din rețele neuronale convoluționale și recurente, rezolvă această sarcină. Pentru grafia latină și cea chirilică veche se folosește o abordare mixtă, care comută între diferite modele de recunoaștere bazate pe calitatea vizuală a textului. Acest lucru ajută la îmbunătățirea semnificativă atât a vitezei, cât și a preciziei de recunoaștere optică a caracterelor, inclusiv și din documentele istorice [27].

Există două metode de antrenare a modelelor OCR bine cunoscute în acest domeniu: antrenare pe date sintetice (imagini generate din textul electronic existent și fonturile disponibile pe computer); sau antrenare pe date reale (perechi formate din forme de glifă sau caracterul imagine și transcripția acestuia - caracterul Unicode) [28]. Prima metodă evită necesitatea de a genera date cu adevăr de bază care sunt necesare pentru a stabili legătura dintre formele de glife și caractere Unicode în timpul procesului de instruire, și, de asemenea, nu este nevoie să preproceseze imaginile reale. Întrucât întregul proces de instruire poate fi automatizat, această metodă de

instruire este cea preferată ori de câte ori este aplicabilă. Cu toate acestea, multitudinea de fonturi istorice nu poate fi comparată cu fonturile existente, iar neregularitățile spațiilor dintre cuvinte, calitatea inferioară a imaginii scanate, petele de uzură, etc. în astfel de documente duc la rezultate de recunoaștere inferioare în comparație cu antrenarea pe date reale [26]. Instruirea OCR pentru documentele istorice trebuie, așadar, să se bazeze pe un proces care utilizează date reale, ceea ce înseamnă că exemplele de antrenare preluate direct din documentele tipărite devin o resursă cheie. Corpusurile istorice cu astfel de exemple, care ar putea servi drept date de instruire pentru fonturile istorice, nu sunt încă disponibile în cantitate suficientă (după cum este menționat mai sus, doar grupuri mici de specialiști se preocupă cu aceasta). O altă problemă o constituie variabilitatea. Tipografiile vechi se caracterizează printr-o diversificare mai mare a fonturilor, deoarece procesul de proiectare, producere și distribuire a tipurilor de litere metalice nu devenise încă o profesie proprie, iar tipografiile vechi au trebuit să producă propriile lor garnituri de litere, ducând la o mare varietate de fonturi istorice [29-30]. Prin urmare, este problematic să antrenezi asocierile dintre glifele tipărite și caracterele UNICODE pe care le reprezintă, utilizând datele de la o tipografie și aplicându-le la o alta.

În lucrarea [26] autorii observă două probleme referitoare la aplicarea metodelor OCR pe documente istorice tipărite. Mai întâi trebuie să fie antrenat un model individual pentru o anumită carte cu tipografia sa specifică. Acest lucru poate fi realizat prin transcrierea unei porțiuni (una sau mai multe pagini) din textul documentului tipărit, care necesită de obicei cunoștințe lingvistice. În al doilea rând, chiar dacă acest model funcționează bine pentru cartea pe care a fost instruit, în mod normal nu produce rezultate bune OCR pentru alte cărți, chiar dacă fonturile lor arată similar. Această barieră tipografică trebuie depășită pentru a folosi metodele OCR în mod eficient în construirea unui corpus istoric. În cele ce urmează autorii descriu experimentele care abordează ambele probleme. Pentru început, se descrie o procedură pentru antrenarea modelelor individuale, folosind un nou algoritm de recunoaștere bazat pe rețele neuronale recurente, așa cum este implementat în motorul OCR *OCRopus* [31]. În calitate de material pentru instruire autorii folosesc documentele scanate din corpusul *RIDGES* [32]. Modelul individual este antrenat pe un singur document cu fonturile sale specifice și, prin urmare, adaptat în mod optim la această carte. Aceste modele produc rezultate excelente de recunoaștere pentru paginile încă nevăzute (neprocesate) ale cărților în baza cărora au fost instruite, dar, din păcate, dau rezultate, în mare parte, slabe, fiind aplicate pe orice alte documente cu font fie și similar. În continuare, autorii explorează viabilitatea antrenării modelelor mixte pe o gamă de tipografii diferite, punând în comun materialul de instruire dintr-o varietate de cărți, cu speranța că aceste modele sunt capabile să generalizeze mai bine procesul de recunoaștere pentru cărți, din care nu au fost preluate exemple de caractere în setul de

antrenare. Toate modelele, chiar și cele individuale, sunt testate doar pe seturi de testare care nu se suprapun cu datele de instruire, folosite în învățarea modelului. Autorii au antrenat două modele mixte pe două părți ale corpusului, partea 1 și partea 2 (Figura 1.1). Fiecare parte conține documente din intervalul de 400 de ani, astfel încât să se poată testa cât de bine se generalizează un model mixt antrenat dintr-un eșantion de cărți pe o gamă largă de date de tipărire cu alte cărți neprocesate care datează din aceeași perioadă. Rezultatul aplicării fiecăruia dintre cele două modele mixte la cărțile din celălalt subset este prezentat în Figura 1.1 (coloanele galbene) împreună cu precizia modelelor individuale (coloanele albastre). Sunt prezentate rezultatele atât în raport cu acuratețea caracterelor (panoul de sus), cât și acuratețea cuvintelor (panoul de jos). După cum era de așteptat, modelele mixte oferă o precizie mai mică în comparație cu modelele individuale, deoarece modelele mixte nu au fost instruite pe cărțile celeilalte părți la care au fost aplicate. Dar, cu excepția a două cărți, ambele modele mixte ating în mod constant o precizie de peste 90% a caracterelor pe cărți care nu au contribuit la setul de antrenare. Într-un caz, modelul mixt prezintă o performanță puțin mai bună decât modelul individual (cazul *1774-U* din Figura 1.1). Această carte este suficient de asemănătoare din punct de vedere tipografic cu cărțile celeilalte părți pe care a fost antrenat modelul mixt, astfel încât recunoașterea este îmbunătățită. Comparând diferența de precizie a caracterelor și a cuvintelor, autorii observă faptul că precizia îmbunătățită a caracterelor obținute din trecerea de la un model mixt la un model individual îmbunătățește, de asemenea, și acuratețea cuvintelor, dar cu o marjă absolută mai mare (aproximativ, reducând la jumătate erorile de caractere), de asemenea, se reduc la jumătate erorile de cuvinte (vezi Tabelul 1.1). Erorile OCR observate atât la recunoașterea modelului mixt, cât și al celui individual, constau dintr-un număr egal de erori de înlocuire (un caracter este confundat cu un alt caracter, adesea similar, cum ar fi „e” și „c”) și erori de ștergere sau inserare (caracterele fie că nu sunt recunoscute deloc, fie sunt introduse caractere false acolo unde nu au fost tipărite). Cel mai frecvent caracter șters sau inserat este spațiul, care atunci când este inserat duce la cuvinte divizate în două sau mai multe fragmente sau, dacă este șters, duce la cuvinte concatenate, în care două sau mai multe cuvinte tipărite sunt recunoscute ca un singur cuvânt. Acest lucru subliniază din nou necesitatea de a dezvolta un model de recunoaștere pe documente reale, deoarece un model adecvat poate fi antrenat doar în acest fel.

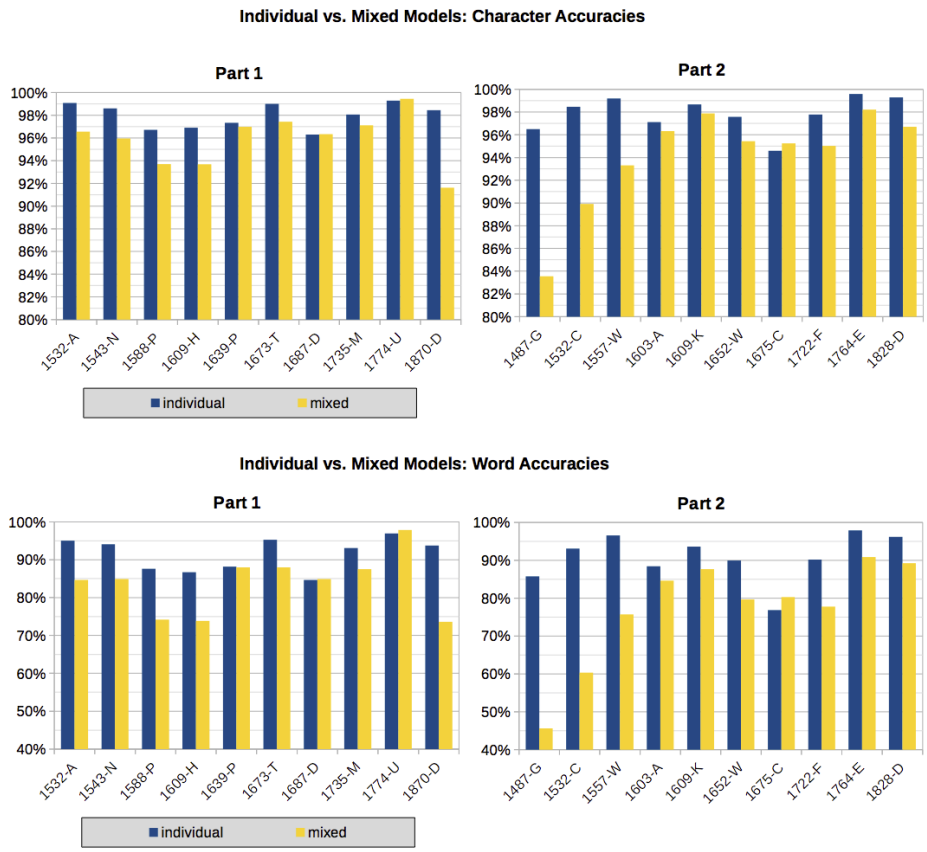



**Figura 1.1. Acuratețea rezultatelor OCR pentru mai multe modele individuale (albastru) și pentru un model mixt (galben) [26].**

**Tabelul 1.1. Acuratețea medie pentru modele individuale și mixte [26].**

| acurateţe medie | Partea 1 | Partea 2 |
|---|---|---|
| *acuratețea la nivel de caractere* | | |
| modele individuale | 98.07% | 97.94% |
| model mixt | 95.81% | 94.27% |
| *acuratețea la nivel de cuvinte* | | |
| modele individuale | 91.53% | 90.71% |
| model mixt | 83.20% | 76.09% |

Prin urmare, instruirea modelelor mixte pare a fi o modalitate de a depăși bariera tipografică și poate oferi modele care se generalizează bine dintr-o gamă largă de cărți [26]. Textul ce rezultă după OCR poate fi luat cel puțin drept o primă aproximare în baza căreia pot fi antrenate modele mai bune utilizând o versiune cu erorile corectate ale textului recunoscut. La aceeași concluzie s-a ajuns într-un studiu pe douăsprezece cărți latine tipărite între 1471 și 1686 în fonturi

Antiqua, efectuat în lucrarea [33], unde este prezentată o metodă pentru a construi modele individuale mai bune cu un efort manual minim, pornind de la un model mixt.

Mai multe lucrări asupra recunoașterii (OCR) documentelor istorice s-au concentrat în mare parte pe *Tesseract* [34], motor OCR care poate fi antrenat pe imagini artificiale generate din fonturi de computer. Tesseract este, fără îndoială, unul dintre cele mai populare motoare OCR open source. A fost creat inițial la Hewlett-Packard între 1985 și 1994 și a fost plasat ca soft open source în 2005. Informații mai detaliate despre capacitățile și istoria dezvoltării Tesseract sunt descrise în lucrarea [35]. Tesseract este încă în curs de dezvoltare activă de către Google și este disponibil în prezent în versiunea 5.1.0[3], conținând inclusiv un nou motor bazat pe rețele neurale cu arhitectura specifică clasei rețelelor neurale recurente. Ambele motoare au puncte forte și puncte slabe și, prin urmare, sunt aplicabile în diferite scenarii de utilizare. Cu toate acestea, antrenarea pe date reale s-a dovedit a fi dificilă și a solicitat anumite eforturi pentru a reconstrui fontul istoric original din glife decupate. Acest lucru a fost realizat și de echipa din Poznań (Polonia) în lucrarea [36] cu instrumentul *Franken+*[4]. Autorii au raportat că acest instrument a reușit să atingă o acuratețe a caracterelor de aproximativ 86% pe colecția de documente ECCO [37].

O abordare complet diferită a fost adoptată cu motorul *OCR Ocular* [38], care este capabil să convertească textul tipărit în text electronic într-un mod complet nesupravegheat (adică, nu este nevoie de un set de date cu *adevăr de bază*). Aceasta poate fi o alternativă viabilă pentru antrenarea modelelor individuale cu efort manual redus, dar pare să consume foarte multe resurse și să fie lentă (transcrierea a 30 de rânduri de text durează 2,4 minute). Rezultatele sale sunt mai bune decât Tesseract și ABBYY FineReader (fără antrenare), dar rămâne de demonstrat că pot atinge în mod constant performanța la recunoașterea caracterelor mai mare de 90%.

În lucrarea [39] este arătat modul în care acuratețea de recunoaștere pentru textele istorice poloneze poate fi îmbunătățită semnificativ prin antrenarea motorului OCR utilizat. Raportul arată îmbunătățiri de la 45 la 80% de acuratețe la recunoașterea caracterelor și 15–55% acuratețe de recunoaștere a cuvintelor, pentru recunoașterea documentelor gotice după antrenarea ABBYY FineReader pe doar câteva pagini. Pentru a valorifica instruirea OCR, sistemele de genul ABBYY FineReader au facilități de bază încorporate pentru a adăuga cel puțin câteva simboluri noi la un limbaj existent [40].

Tesseract și alte programe open-source, pe de altă parte, oferă mai multe posibilități de instruire. Există diverse lucrări despre modul în care au fost abordate problemele specifice de

[3] https://github.com/tesseract-ocr/tesseract/releases/tag/5.1.0
[4] https://emop.tamu.edu/outcomes/Franken-Plus

instruire OCR. Un exemplu [41] arată cum Tesseract a fost antrenat pentru a recunoaște greaca veche. Procesul descris se bazează în principal pe scripting și pe unele intervenții manuale. Autorii oferă câteva recomandări generale, dar nici o abordare sistematică legată de alegerea parametrilor și setărilor nu este descrisă, nefiind prezentate nici justificările statistice. Un alt exemplu poate fi găsit în [42], unde autorii antrenează motorul Tesseract să recunoască *Odia* – un script indian. Aici, setul de antrenare este mai întâi generat într-un mod artificial și apoi introdus în instrumentele standard de linie de comandă Tesseract. Deși acest lucru ar putea funcționa pentru limbile (seturile de caractere) care sunt cunoscute în principiu, nu ar fi o opțiune viabilă dacă se așteaptă să apară caractere și simboluri necunoscute (ceea ce este destul de des întâlnit în documentele istorice).

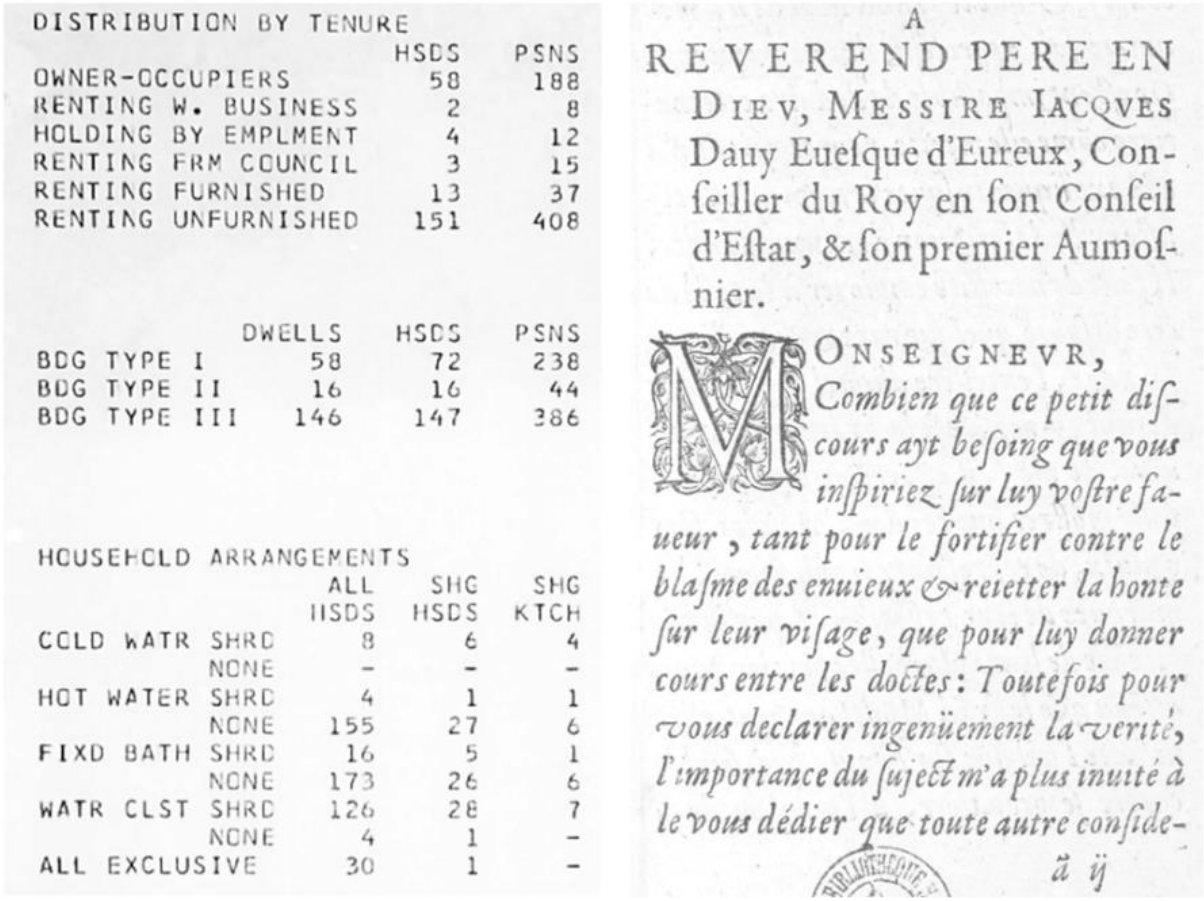
În lucrarea [43] autorii descriu o abordare eficientă pentru instruirea motoarelor OCR folosind sistemul de analiză a documentelor *Aletheia* [44]. Dezvoltarea sistemului Aletheia a început inițial în 2001 în scopul creării unui sistem de producere de seturi de date cu *adevăr de bază* pentru recunoașterea structurii paginii și a blocurilor de text. De atunci, Aletheia a evoluat într-un sistem complet de analiză a imaginii documentelor, încorporând suport pentru mai multe formate de fișiere, procesare/îmbunătățire a imaginii și metode de corecție geometrică, OCR integrat, funcții de introducere și manipulare a blocurilor de text și altele. Arhitectura generală a sistemului Aletheia are un design modular care permite integrarea diferitelor motoare OCR. Motorul Tesseract este disponibil în mod implicit în Aletheia. Comunicarea dintre module se bazează pe o interfață de linie de comandă, iar comunicarea de date se face prin PAGE XML (existând și compatibilitatea și cu alte formate precum ALTO și FineReader XML) [45]. Toate componentele necesare pentru antrenarea modelelor OCR sunt integrate în Aletheia: pregătirea datelor de antrenare, procesele de antrenare ale modelului OCR în sine, recunoașterea textului și evaluarea modelului antrenat. Un astfel de sistem de instruire și evaluare, ghidat printr-o interfață grafică cu utilizatorul, permite formarea incrementală iterativă pentru a obține cele mai bune rezultate. Eficiența sistemului propus în [43] este bazat pe fluxul de lucru integrat pe care îl permite. Eficacitatea sistemului este demonstrată prin rezultatele experimentelor pe diferite seturi de date și scenarii de aplicare, inclusiv prezența a mai mult de un font în același document. Pe lângă descrierea detaliată a sistemului de antrenare al motorului OCR propus, această lucrare raportează și un număr de experimente efectuate pe diferite seturi de date pentru a cerceta condițiile ideale de antrenare în ceea ce privește dimensiunea și calitatea unui set de antrenare. Autorii au validat eficacitatea procesului de instruire utilizând două seturi de date foarte diferite, fiecare reprezentând un scenariu de utilizare realist, în care antrenarea motorului OCR poate face o diferență: un set de date din Recensământul din 1961 pentru Anglia și Țara Galilor (asemănătoare cu fonturi mai „moderne” mono-spațiate) și o carte de la Biblioteca Națională Franceză tipărită în

anul 1603; datele fiind colectate și fundamentate pentru proiectul IMPACT [46]. În Figura 1.2 autorii prezintă două exemple, iar Tabelul 1.2 înfățișează caracteristicile și dimensiunile seturilor de antrenare și de testare. Toate experimentele au fost realizate prin crearea unui set inițial de exemple de antrenare (glife împreună cu adevărul de bază), iar rezultatele au fost evaluate folosind o metrică bazată pe text. Procesul de antrenare a fost realizat prin Aletheia. Caracterele speciale și ligaturile au fost introduse așa cum le putem vedea pe pagină (*transcriere diplomatică*[5]) și nu transliterate. Strategiile luate în considerare de autori pentru selectarea exemplelor de antrenare au implicat: eliminarea glifelor parțial șterse/deformate și eliminarea glifelor cu aspect similar. Impactul fiecărei strategii a fost testat prin eliminarea treptată a tot mai multor exemple de antrenare din setul de date. Pentru a măsura calitatea rezultatelor OCR a fost folosit un instrument numit *TextEval* [47]. Printre alte măsuri, se folosește o implementare a măsurii Universității din Nevada [48], care se bazează pe distanța de editare a șirurilor. Calitatea evaluării este raportată în procente, unde 100% indică faptul că textul a fost recunoscut perfect. Pentru a măsura impactul modelelor antrenate, au fost comparate rezultatele OCR cu și fără antrenare. *Fără antrenare,* în acest context, înseamnă folosirea fișierelor de informații lingvistice implicite furnizate de motoarele OCR. Pentru caracterul complet al comparației, autorii au evaluat și sistemul comercial ABBYY FineReader Egine 11. Trebuie remarcat faptul că setul de date *Recensământ* include doar caractere latine majuscule (precum și cifre și semne de punctuație). Tabelul 1.3 prezintă rezultatele evaluării realizate de către autori. Motorul Tesseract antrenat depășește toate celelalte configurații, iar autorii au izbutit să investigheze acuratețea pentru motorul FineReader *cu antrenare*. Pentru ambele seturi de testare, există o creștere a performanței de aproximativ 5% față de rezultatele Tesseract OCR fără antrenare.

**Tabelul 1.2. Seturile de date pentru evaluarea modelelor în [43].**

| Set de date | Caractere/font | Set de antrenare | Set de testare |
|---|---|---|---|
| Recensământ din 1961 | 40 de clase de caractere, litere mari latine, cifre și semne de punctuație; un singur font | 2150 de glife | 2115 glife |
| Biblioteca Națională Franceză (1603) | 74 de clase de caractere; Latină cu caractere și ligaturi franceze, două fonturi | 773+1321 glife | 2040 glife |

[5] Transcrierea diplomatică încearcă să reprezinte totul exact așa cum se vede într-un document, fără modificări, precum extinderea abrevierilor, transliterarea numelor proprii etc (https://libguides.hull.ac.uk/archival-skills/transcription).



DISTRIBUTION BY TENURE

| | HSDS | PSNS |
|---|---|---|
| OWNER-OCCUPIERS | 58 | 188 |
| RENTING W. BUSINESS | 2 | 8 |
| HOLDING BY EMPLMENT | 4 | 12 |
| RENTING FRM COUNCIL | 3 | 15 |
| RENTING FURNISHED | 13 | 37 |
| RENTING UNFURNISHED | 151 | 408 |

| | DWELLS | HSDS | PSNS |
|---|---|---|---|
| BDG TYPE I | 58 | 72 | 238 |
| BDG TYPE II | 16 | 16 | 44 |
| BDG TYPE III | 146 | 147 | 386 |

HOUSEHOLD ARRANGEMENTS

| | | ALL HSDS | SHG HSDS | SHG KTCH |
|---|---|---|---|---|
| COLD WATR | SHRD | 8 | 6 | 4 |
| | NONE | - | - | - |
| HOT WATER | SHRD | 4 | 1 | 1 |
| | NONE | 155 | 27 | 6 |
| FIXD BATH | SHRD | 16 | 5 | 1 |
| | NONE | 173 | 26 | 6 |
| WATR CLST | SHRD | 126 | 28 | 7 |
| | NONE | 4 | 1 | - |
| ALL EXCLUSIVE | | 30 | 1 | - |

A
REVEREND PERE EN
DIEV, MESSIRE IACQVES
Dauy Eueſque d'Eureux, Conſeiller du Roy en ſon Conſeil d'Eſtat, & ſon premier Aumoſnier.

MONSEIGNEVR,
*Combien que ce petit diſcours ayt beſoing que vous inſpiriez ſur luy voſtre faueur, tant pour le fortifier contre le blaſme des enuieux & reietter la honte ſur leur viſage, que pour luy donner cours entre les doctes: Toutefois pour vous declarer ingenüement la verité, l'importance du ſuject m'a plus inuité à le vous dédier que toute autre conſide-*

ä ij

**Figura 1.2. Exemple de pagini ale setului de date de evaluare din lucrarea [43].**

**Tabelul 1.3. Acuratețea OCR cu și fără antrenare în [43].**

| Motoare și modele OCR | Acurateţe OCR după setul de date (%) | |
|---|---|---|
| | Recensământ din 1961 | Biblioteca Națională Franceză (1603) |
| FineReader (fără antrenare) | 88.40 | 78.09 |
| Tesseract (fără antrenare) | 90.08 | 84.93 |
| Tesseract (cu antrenare) | 95.40 | 90.14 |

Dacă ar fi să vorbim despre instrumentele OCR gratuite, atunci cele mai populare instrumente sunt *OCRopus*[6] sau *Ocropy, Kraken*[7]*, Tesseract*[8] și *Calamari*[9]. Ocropy și Kraken

[6] https://github.com/ocropus
[7] https://github.com/mittagessen/kraken
[8] https://github.com/tesseract-ocr/tesseract
[9] https://github.com/Calamari-OCR/calamari

antrenează o rețea neurală cu arhitectura *LSTM*[10] cu un singur strat de neuroni, iar noile versiuni ale Tesseract și Calamari antrenează modele OCR folosind de asemenea învățarea profundă, dar cu rețele multistrat de tip CNN[11] și LSTM.

În lucrarea [49] se aplică diferite metode OCR cu Ocropy pe documente istorice tipărite cu font latin și se obține o acuratețe bună. Autorii lucrării [50] prezintă arhitectura Ocropy și explică diferiți pași ai unui proces OCR. Springmann și Lüdeling în lucrarea [28] (analizată mai sus) folosesc Ocropy pentru a recunoaște documentele tipărite între 1487 și 1870 și raportează o performanță la nivel de caractere mai mare de 90%.

În lucrarea [51] autorii prezintă pentru prima dată softul numit Calamari (numit și Calamari OCR) - un set de instrumente pentru instruirea și recunoașterea liniilor (rândurilor) de text din imagine. A fost creat ca o variantă îmbunătățită în comparație cu Ocropy. Calamari acceptă o arhitectură de rețele neuronale multistrat CNN-LSTM definită de utilizator, care s-a dovedit că mărește acuratețea modelului. Ei folosesc Tensorflow ca backend, ceea ce se pare că crește performanța de calcul în comparație cu Ocropy, mai ales în timp ce se antrenează și se recunoaște pe un GPU. Instrumentul Calamari poate fi folosit ca înlocuitor pentru Ocropus și oferă și alte caracteristici importante. De exemplu, modelele pot fi antrenate și textul poate fi recunoscut pe un GPU, ceea ce îmbunătățește performanța. Au fost implementate funcții suplimentare, cum ar fi *oprirea timpurie*[12] a procesului de antrenare, *validarea încrucișată*[13] și *preantrenarea*[14]. Toate aceste caracteristici duc la rate mai mici de eroare [53]. În lucrarea [54], performanța lui Calamari este testată în comparație cu Ocropy pe documente istorice, demonstrând că o combinație a unei rețele convoluționale și a unei rețele LSTM are performanțe mai bune decât un singur strat LSTM (folosit în Ocropy). S-a constatat că pentru recunoașterea unei cărți un model pre-antrenat necesită exemple de antrenare cu adevăr de bază din 60 de rânduri de text pentru a obține o rată de eroare sub 2%. Autorii verifică atât viteza antrenării cât și cea a recunoașterii. Antrenarea rețelei neurale din Calamari este mai rapidă decât antrenarea Ocropy atunci când sunt utilizate mai multe nuclee

[10] Long short-term memory (LSTM) este o rețea neuronală artificială recurentă utilizată în domeniile inteligenței artificiale și învățării profunde. O astfel de rețea neuronală (recurentă) poate procesa nu numai puncte de date individuale (cum ar fi imagini), ci și secvențe întregi de date (cum ar fi audio sau video).

[11] O rețea neurală cu convoluții (cunoscută și sub numele de CNN sau ConvNet) reprezintă o categorie de rețele neurale artificiale, utilizată în mod predominant pentru procesarea și identificarea imaginilor [52].

[12] În învățarea automată, oprirea timpurie este o formă de regularizare utilizată pentru a evita supraînvățarea unui model atunci când acesta este învățat utilizând o metodă iterativă, cum ar fi coborârea gradientului (gradient descent).

[13] Validarea încrucișată [56-58] sau testarea în afara setului de antrenare reprezintă tehnici de validare a modelului de învățare automată pentru evaluarea modului în care rezultatele modelului antrenat se vor generaliza la un set de date independent.

[14] Preantrenarea funcționează după cum urmează. Este dat modelul de învățare automată *m* și un set de date *A* pe care se antrenează *m*. De asemenea, este dat un set de date *B*. Înainte de a începe antrenarea modelului *m* pe *B*, *m* este (pre)antrenat pe *A*.

CPU (mai mult decât 4). Cu toate acestea, antrenarea unui model pe un GPU este de 4 ori mai rapidă. În faza de predicție, Calamari este mai rapid de 3 ori decât Ocropy chiar și cu un singur nucleu CPU și aproximativ de 30 de ori mai rapid pe un GPU [54].

În lucrarea [53] autorii folosesc instrumentul *Calamari* pentru a recunoaște un corpus de ziare istorice publicate în Finlanda între anii 1771–1929 [55]. De menționat este faptul că acest corpus a fost recunoscut anterior cu ABBYY FineReader 11 și prezintă o acuratețe la nivel de caractere între 87% și 92%, prin urmare, rata de eroare este destul de mare pentru analizarea lingvistică calitativă a corpusului, autorii considerând și nevoia de a recunoaște repetat întreg corpusul de documente utilizând avantajele instrumentarului din Calamari. Acest corpus conține date foarte diverse scrise într-un limbaj non standard [53]. Ziarele din Finlanda din secolul al XVIII-lea până la începutul secolului al XX-lea au fost tipărite în două limbi de bază ale Finlandei (finlandeză și suedeză) folosind două familii de fonturi diferite: *gotice* (Blackletter) și *Antiqua*. Vom remarca că datele nu sunt distribuite uniform. În documentele mai vechi, există mai mult material tipărit în suedeză cu fonturi gotice, în timp ce documentele moderne sunt tipărite în mare parte în finlandeză cu fonturi Antiqua. Cu toate acestea, există perioade în care ambele limbi și ambele familii de fonturi au fost utilizate pe scară largă, uneori chiar și pe aceleași pagini de ziare. Standardizarea limbii literare finlandeze a început în secolul al XIX-lea [59]; prin urmare, o mare parte din corpus conține ortografii din diferite dialecte finlandeze [43]. Corpusul este foarte mare, cu aproape 5 miliarde de token[15]-uri, așa că autorii încearcă să găsească totodată și o metodă eficientă în timp pentru recunoașterea repetată a corpusului. Modalitatea OCR propusă de autori include preprocesarea imaginii, unde procesul de bază îl constituie convertirea în alb-negru; segmentarea imaginii în linii de text (exemplele de antrenare constau din linii de text și secvențe de caractere în sine − foarte important aici este această diferență de abordare, în raport cu motoarele OCR care se antrenează cu exemple de învățare la nivel de glifă, cum ar fi Tesseract 3 sau ABBYY FineReader. Am cădea de acord că este dificil să segmentezi corect fiecare glifă, astfel se produc multe greșeli de segmentare, iar crearea șabloanelor de antrenare la nivel de glifă pentru fonturi diferite, în special pentru documente istorice, necesită foarte mult timp. Capacitatea de a prezenta rânduri întregi de text în rețelele neuronale LSTM, segmentarea la nivel de linie afirmându-se drept cea mai recentă tehnologie [53]. În continuare, în procesul de recunoaștere are loc pregătirea setului de date și antrenarea unui model de recunoaștere, iar în calitate de rezultat se obține textul

[15] Un token este o instanță a unei secvențe de caractere dintr-un anumit document, care sunt grupate împreună ca o unitate semantică utilă pentru procesare. Token-urile sunt adesea denumite în mod liber ca *termeni* sau *cuvinte.* Sarcina de a segmenta un text în token-uri, aruncând anumite caractere, cum ar fi semnele de punctuație, se numește *tokenizare.*

care poate fi corectat folosind metode lingvistice de postprocesare. Rezultatul final este de obicei înscris în fișiere XML. La pregătirea setului de antrenare și testare sunt utilizate colecții de date deja disponibile [60], care constituie circa 9500 de linii de text finlandez și 6500 de linii de text suedez (câte 418 din fiecare set sunt folosite doar pentru testare), precum și seturi de date separate, create de către autori. Ambele seturi de date au fost culese aleatoriu din corpusul de ziare și reviste istorice [55]. Setul de date finlandez este extras din perioada 1820–1939, iar setul de date suedez - din 1771 până în 1874. Pe lângă seturile de date existente, autorii au adăugat 5000 de linii din documente suedeze și 4000 finlandeze din același corpus și le-au transcris manual. Pentru a obține exemple de antrenare din corpus, s-au folosit informațiile de segmentare și OCR ale ABBYY FineReader, stocate în fișiere *METS-ALTO*[16]. În urma pregătirii setului de date autorii au obținut aproximativ 11.500 de rânduri de text în suedeză și 13.500 în finlandeză, ambele constând dintr-un procent similar de linii de fonturi gotice și *Antiqua*. Seturile de date de testare pentru finlandeză și suedeză, în particular, conțin fiecare câte 418 linii de text. Pentru a testa rezultatele modelului OCR, autorii folosesc validarea încrucișată pe setul de date împărțit în 5 părți de dimensiuni egale. Pentru evaluarea rezultatelor se efectuează următoarele măsurări: rata de eroare a caracterelor (CER) și rata de eroare a cuvintelor (WER). Rata de eroare a caracterelor este procentul de caractere recunoscute greșit din numărul total de caractere recunoscute. În mod similar, rata de eroare a cuvintelor este numărul de cuvinte eronate împărțit la suma cuvintelor corecte și greșite după recunoaștere. Pentru a obține numărul de erori, autorii au aliniat mai întâi adevărul de bază cu rândurile de text recunoscut la nivel de caractere (atât pentru CER, cât și pentru WER) și au calculat distanța *Levenshtein* [61] dintre acestea. În faza de recunoaștere, autorii lucrării menționate mai sus [53] folosesc instrumentul de predicție din Calamari. Din acest punct de vedere, principalul avantaj al acestui instrument constă în posibilitatea rulării pe un GPU, ceea ce face ca faza de recunoaștere să fie foarte rapidă. O altă caracteristică a instrumentului este capacitatea de a recunoaște o imagine cu mai multe modele în același timp și apoi de a alege rezultatul optim utilizând un *mecanism de votare*. Mecanismul de votare este un *meta* model de învățare automată care combină predicțiile din mai multe alte modele. Este o tehnică care poate fi folosită pentru a îmbunătăți performanța modelului, obținând în mod ideal o performanță mai bună decât orice model. Fiecare model prezice mai mulți candidați cu probabilitățile care le însoțesc și

[16] Standardul METS este o schemă flexibilă pentru descrierea unui obiect digital complex (cum ar fi un număr de ziar digitalizat). METS descrie structura obiectului, dar nu codifică conținutul textual real al obiectului. Standardul ALTO umple acest gol prin codificarea conținutului textual al unei pagini digitizate în detaliu, inclusiv stiluri și machete. Pe lângă codificarea textului digitizat în sine, ALTO codifică coordonatele spațiale ale fiecărei coloane, linii și cuvinte așa cum apar pe pagină [62]. METS și ALTO sunt standarde XML menținute de Biblioteca Congresului SUA (https://www.loc.gov/standards/alto/techcenter/use-with-mets.html).

apoi mecanismul de vot decide care candidat câștigă. Autorii lucrării [55] arată că votul reduce întotdeauna erorile. Dezavantajul metodei îl constituie necesitatea recunoașterii prin utilizarea mai multor modele, ceea ce încetinește procesul de recunoaștere. Pentru a evalua performanța modelelor de recunoaștere, autorii au efectuat experimente cu: modele mixte (modele antrenate atât pe date finlandeze, cât și suedeze); modele monolingve (modele instruite pe date finlandeze sau suedeze); aplicarea mecanismului de votare pe combinații de modele; post-corectare. Modelele mixte realizează o eroare medie de 2,6% CER și 10% WER. Pentru suedeză, în particular, se obține 3,8% CER și 13% WER, iar pentru finlandeză se produce 1.7% CER și 8% WER. Aplicând mecanismul de votare pe modele ce au operat cu setul de testare suedez s-au obținut 2,8% CER și 11% WER, iar setul de testare finlandez a generat cele mai bune rezultate din 5 modele mixte egale cu 1,9% CER și 8% WER. În cele din urmă, autorii efectuează o postprocesare (corectarea erorilor după recunoaștere) asupra noilor rezultate OCR. Rezultatele arată o creștere semnificativă a acurateței, rezultând în 1,7% CER pe setul de testare finlandez și 2,7% CER pe setul de testare suedez. Cea mai mare realizare a autorilor este formarea cu succes a unui model mixt pentru întregul corpus și găsirea unei configurații a mecanismului de vot care să îmbunătățească și mai mult rezultatele.

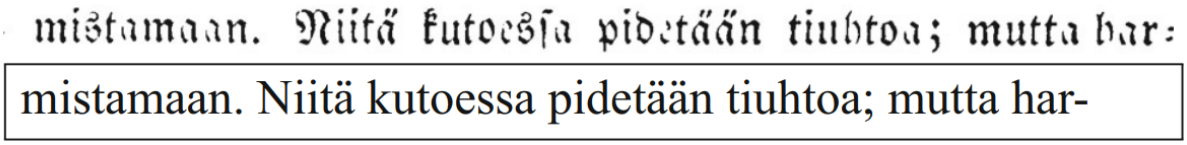



**Figura 1.3. Un exemplu de antrenare cu o linie de text din imagine pusă în corespondență cu secvența de caractere pe care o reprezintă (*adevărul de bază*) din lucrarea [53].**

În continuare vom analiza unele platforme sau cadre de digitizare și procesare a documentelor istorice elaborate în proiecte importante referitoare la digitizarea patrimoniului cultural-istoric. Mai sus am menționat unele metode și instrumente particulare de preprocesare a imaginii, de recunoaștere a caracterelor/liniilor de text, de post-procesare OCR. Desigur că ar fi mai eficient și mai raționalizat procesul de digitizare și procesare a documentelor istorice, dacă tot instrumentarul necesar s-ar afla într-un singur loc, integrat într-o singură aplicație software. Astfel de aplicații ar putea fi numite *platforme* sau *cadre de digitizare*.

În lucrarea [63] se descrie un cadru web complex și flexibil numit *Historical Document Processing and Analysis Framework (HDPA*[17]*)* pentru gestionarea și analiza documentelor istorice, cu accent principal pe OCR. Cadrul conține opt module pentru a facilita trei sarcini

[17] Cadrul HDPA este disponibil gratuit la adresa http://ocr-corpus.kiv.zcu.cz/ (accesat pe 20 iunie 2022).

principale: preprocesarea și segmentarea imaginii, crearea setului de date pentru antrenarea modelului OCR și recunoașterea în sine. Acest cadru este disponibil gratuit, pentru scopuri de cercetare. Autorii demonstrează că acest sistem este eficient și poate economisi munca umană în procesul de pregătire a seturilor de date pentru OCR. Aplicația web este scrisă în Django[18], ceea ce permite dezvoltatorilor și cercetătorilor să elaboreze module Python individuale. Modulele pot rula separat, iar Django are rolul unui hub care conectează interfața cu utilizatorul cu rezultatele modulelor dorite. Cadrul HDPA propus conține unități sau grupuri funcționale care efectuează trei sarcini principale. Primul grup funcțional se preocupă de preprocesarea imaginii și segmentarea paginii. Al doilea grup funcțional oferă instrumente pentru crearea seturilor de date (adevărul de bază) pentru antrenarea modelului OCR. Al treilea grup funcțional cuprinde motorul OCR. Lipsește un modul de postprocesare integrat în mod implicit în cadrul HDPA, dar autorii asigură o integrare ușoară a modulelor noi. În acest fel, utilizatorii pot personaliza cu ușurință sistemul HDPA pentru nevoile lor specifice. Cadrul oferă în prezent utilizatorului două module de preprocesare de bază, și anume binarizarea și rotirea imaginii. Modulul de creare a setului de date de antrenare este util atunci când dorim să antrenăm un nou model OCR. Acesta oferă instrumente care servesc pentru crearea unui set de imagini cu o singură literă/glifă, decupate direct din imaginile încărcate de către utilizator. Din astfel de imagini pot fi compuse propoziții asemănătoare cu cele reale. O altă modalitate de a crea seturi de date sintetice este utilizarea unui instrument de generare a imaginilor cu text. Aici se antrenează un model OCR în două etape: antrenarea pe date sintetice mari, care ajută la învățarea formelor glifelor; antrenarea pe o cantitate mică de imagini cu linii de text decupate din pagini reale, care asigură faptul că modelul poate învăța unele aspecte specifice ale datelor reale, ce nu pot fi generate în liniile sintetice. Pentru a permite crearea seturilor de date cu adevăr de bază pentru date reale autorii oferă un instrument pentru adnotarea liniilor de text. Sistemul OCR propus de autori se bazează pe învățarea automată și folosește rețele CNN pentru extragerea caracteristicilor și o rețea neurală recurentă LSTM bidirecțională pentru recunoașterea secvențială a liniilor de text. În figura 1.4 autorii prezintă arhitectura cadrului HDPA. Arhitectura este modulară, compusă din opt module (*M1–M8* în Figura 1.4) încapsulate în trei unități funcționale (*U1–U3* în Figura 1.4) [63]. Prima unitate funcțională *U1* se ocupă de preprocesarea și segmentarea imaginii. Preprocesarea include transformări importante de imagine și corecții necesare pentru o analiză și segmentare reușită a imaginii. Scopul unității *U1* este de a pregăti linii de text din imagine pentru antrenarea motorului OCR. Modulul *M1* din *U1* efectuează binarizarea imaginii. Binarizarea imaginilor poate fi considerată o problemă

[18] Django este un cadru web de nivel înalt scris în limbajul Python care încurajează dezvoltarea rapidă și oferă un design curat și pragmatic. Este gratuit și open source. https://www.djangoproject.com/

de etichetare a pixelilor [63-64]. Este definită ca o funcție *f* care pune în corespondență intensitățile pixelilor imaginii de la intrare *I* la valorile *0* sau *1* în imaginea binarizată de la ieșire, de forma:

$$f(I) = \begin{cases} 1 & dacă\ I(x, y) >= T \\ 0 & dacă\ I(x, y) < T \end{cases} \qquad (3)$$

unde *T* este valoarea pragului, iar *x* și *y* sunt coordonatele pixelilor. Modulul de binarizare din [63] folosește metoda adaptivă de prag propusă în lucrarea [65]. Metoda clasifică mai întâi conținutul imaginii în mai multe clase și apoi aplică două abordări speciale pentru a determina un prag pentru fiecare pixel. Următorul modul, *M2,* se ocupă de rotirea imaginii la stânga/dreapta astfel încât rândurile de text să fie aliniate pe orizontală. În așa mod segmentarea la nivel de linie va produce rezultate mai bune. Autorii au implementat o metodă bazată pe profiluri de proiecție orizontale (PO) [63, 66]. Această metodă rotește imaginea cu diferite unghiuri într-un interval de la $\theta_{min}$ la $\theta_{max}$ și maximizează o funcție de criteriu *c* calculată din valorile profilului de proiecție. Proiecția orizontală a unei imagini *I* este definită ca:

$$PO(y) = \sum_{x=0}^{w-1} I(x, y) \qquad (4)$$

unde *w* este lățimea imaginii *I, x* și *y* sunt coordonatele pixelilor imaginii *I*, iar funcția de criteriu pentru un unghi arbitrat este calculată de funcția *c*, definită ca:

$$c(\theta) = \sum_{y=1}^{h-1} (PO(y) - PO(y-1))^2 \qquad (5)$$

unde *h* este înălțimea imaginii *I.* Autorii cadrului HDPA au ales această metodă datorită simplității și eficienței sale. În continuare, modulul *M3* detectează și extrage blocuri de text din paginile îndreptate în modulul *M2*. În acest modul, autorii au implementat rețeaua convoluțională U-Net [63, 67] și au antrenat-o pe două seturi de date de antrenare diferite: primul set de date de antrenare a fost cel al proiectului *Europeana* [68], iar al doilea set de date a fost creat din documente istorice din proiectul *Porta fontium* [69], care are drept scop digitizarea documentelor de arhivă din zona de frontieră ceho-bavareză. Un rezultat al modulului *M3* este afișat în Figura 1.5, unde sunt identificate casetele de delimitare și desenate pe imaginea originală. Ultimul modul din unitatea funcțională *U1* este *M4,* care segmentează și extrage liniile de text. Pentru a implementa această

funcționalitate, autorii folosesc ARU-Net [63, 70], o rețea neurală multistrat concepută pentru a detecta liniile de la bază (linia pe care se află literele) în manuscrise. Această rețea neurală poate detecta liniile în paginile cu dimensiune variabilă a fontului, dar pentru liniile de text autorii determină dimensiunea fontului, unde folosesc un algoritm bazat pe un profil de proiecție al unei regiuni deasupra liniei de bază detectate. În funcție de profil poate fi calculată aproximativ înălțimea *x* a fontului. Apoi, adăugând înălțimea ascendenților și descendenților, se identifică marginile liniei de text și se decupează linia. În unele cazuri pot fi decupate și părți ale liniilor învecinate. O vizualizare a procesului de segmentare a liniilor de text este prezentată în Figura 6. Următoarele 3 module (*M5, M6, M7*) fac parte din unitatea funcțională *U2*. Această unitate funcțională este utilizată pentru crearea/generarea datelor de antrenare pentru modelul OCR. Pentru a antrena un motor OCR cu linii de text este nevoie de imagini cu linii de text cu etichete corespunzătoare (adevărul de bază). Astfel de seturi de date pot fi obținute prin două moduri: 1 – generarea de date sintetice; 2 – adnotarea imaginilor cu linii de text extrase din imaginea originală. Acest cadru permite crearea a două tipuri de date sintetice separate. Primul tip de date poate fi creat prin punerea laolaltă a imaginilor cu caractere individuale (decupate din imaginile documentului original) conform unui text semnificativ. Acest lucru se face în modulul *M5* unde se segmentează o linie de text în caractere individuale pentru a pregăti mai multe reprezentări diferite pentru fiecare caracter. Apoi, conform unui text istoric, în modulul *M6* se generează imagini cu linii de text prin concatenarea imaginilor caracterelor extrase în *M5*. Ultimul modul *M7* este folosit pentru a adnota linii întregi extrase din imaginea originală. Dacă nu este disponibil niciun model OCR, ne limităm doar la o simplă interfață grafică de adnotare care permite utilizatorului să completeze textul redat în imagine. După adnotarea unui set inițial de linii de text, este posibil de antrenat un model OCR în modulul *M8*. Unitatea *U3* conține modulul *M8*, care este însuși motorul OCR. Motorul utilizează o abordare bazată pe procesarea liniilor de text, care recunoaște imaginile liniilor de text extrase și generează secvența de caractere prezisă. Modulul *M8* se ocupă atât de faza de antrenare, cât și de faza de recunoaștere. Dezvoltatorii cadrului HDPA au inclus în modulul *M8* modele preantrenate pe un set de date sintetice care conține 25,000 de imagini cu linii de text. Textele utilizate la generarea imaginilor sintetice se bazează pe documente vechi germane pentru a se asigura că limbajul corespunde cu cel folosit în documentele prelucrate de autori. De asemenea, aceștia au folosit setul de date *Porta fontium* adnotat la antrenare. Motorul OCR propus în modulul *M8* utilizează o combinație de rețea neuronală convoluțională și recurentă. CNN este folosit pentru extragerea caracteristicilor, în timp ce LSTM este folosit pentru recunoașterea în sine. Arhitectura este o versiune simplificată a rețelei propuse în lucrarea [71]. Pentru evaluarea rezultatelor OCR, autorii folosesc acuratețea/precizia medie la nivel de linii de

text, WER și CER bazate pe rezultatele validării încrucișate pe 10 pagini de testare împărțite în 5 părți egale. Amintim că WER este rata erorii la nivel de cuvinte, iar CER este rata erorii la nivel de caractere. În cazul lucrării [63], precizia medie indică câte imagini cu linii de text din toate cele procesate au fost recunoscute corect, autorii obținând rezultatul de 0.488. Rezultatele pentru WER și CER sunt *0.11* și respectiv *0.024*. Autorii menționează că dezvoltarea viitoare a cadrului se va îndrepta către construirea unei extensii, care va permite aplicarea metodelor de procesare a limbajului natural asupra datelor transcrise, ce va include astfel de instrumente precum recunoașterea entităților numite, clasificarea și căutarea inteligentă a textului integral în documente.

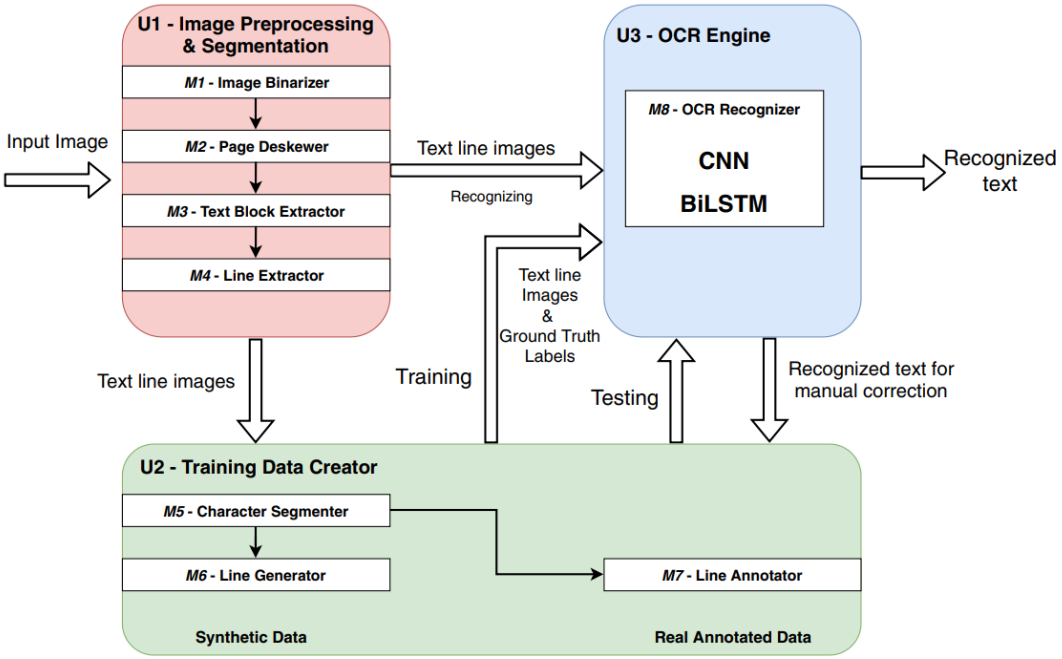



**Figura 1.4. Arhitectura cadrului HDPA [63].**


**Figura 1.5. Rezultatul procesării imaginii cu modulul *M3* din HDPA [63].**

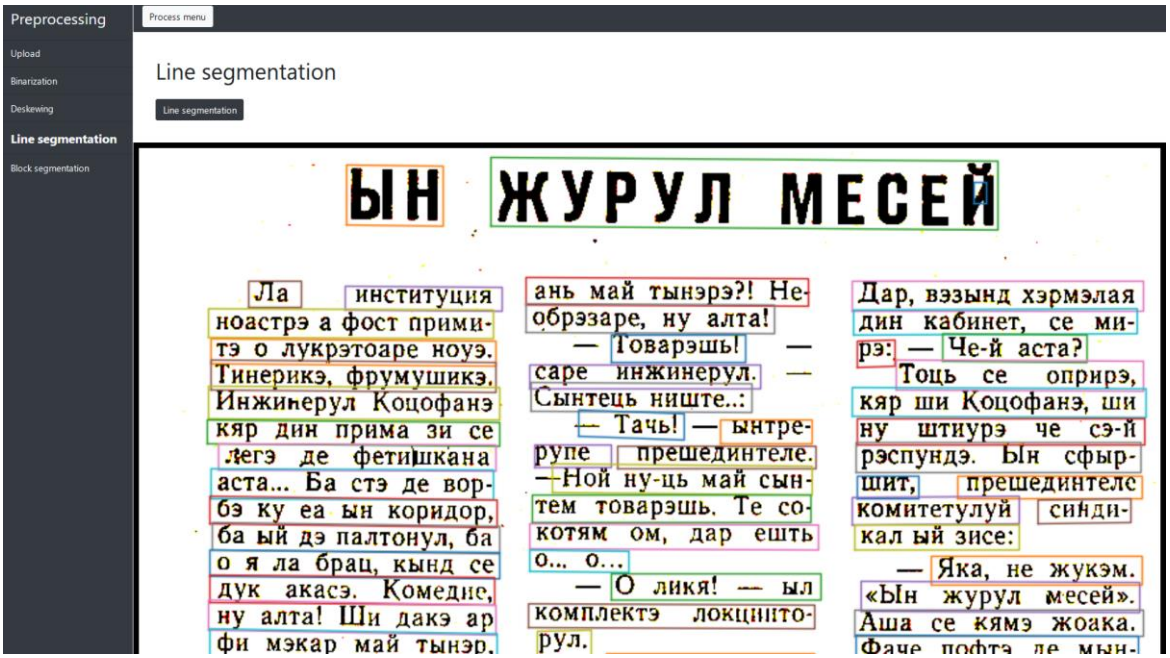



**Figura 1.6. Rezultatul procesării imaginii cu modulul *M4* din HDPA.**

Vorbind în continuare despre platformele de digitizare, procesare și analiză a documentelor istorice vom menționa în mod repetat și în acest context platforma *Aletheia*. În Aletheia, o focalizare deosebită a autorilor este pe analiza aspectului paginii documentului și segmentarea paginii. Segmentarea documentelor sau analiza aspectului documentului este procesul de identificare și clasificare a regiunilor de interes într-o imagine scanată a unui document text. Un

sistem de citire necesită delimitarea zonelor (blocurilor) de text de cele non textuale și redarea în ordine corectă a citirii lor [72]. Detectarea și etichetarea diferitelor blocuri, precum blocuri de text, blocuri de ilustrații, simboluri matematice și tabele încorporate într-un document se numește analiză de aspect geometric [73]. Blocurile de text joacă roluri logice diferite în interiorul documentului (titluri, subtitrări, note de subsol etc.), iar acest tip de etichetare semantică este domeniul de aplicare al analizei aspectului logic. Aletheia poate detecta automat obiecte pe patru niveluri: regiuni de interes (text, tabele, formule, note muzicale etc), linii de text, cuvinte și glife. Contururile obiectelor pot fi ajustate de utilizator. Crearea adevărului de bază este o altă caracteristică a platformei Aletheia. Adevărurile de bază sunt stocate în formatul PAGE XML [74].

O altă platformă de digitizare este *Transkribus* [75] – un instrument complex dezvoltat în cadrul *proiectului READ* [76] de la Universitatea din Innsbruck, care se ocupă de recunoașterea, transcrierea și căutarea documentelor istorice. Oferă o serie de instrumente pentru procesarea automată a documentelor istorice, cum ar fi recunoașterea textului scris de mână, analiza aspectului paginii, înțelegerea documentelor, identificarea scriitorului sau recunoașterea optică a caracterelor (OCR). Pentru OCR Transkribus utilizează motorul ABBYY Finereader. Transkribus nu are suport pentru generarea oricărui tip de date sintetice, o caracteristică bine implementată în cadrul HDPA [63]. În Germania recent a apărut proiectul *OCR-D* [77], cu 8 module speciale axate pe diverse etape ale OCR. Împreună cu acest proiect vine platforma *OCR4all* [78], instrument open-source care oferă un flux de lucru OCR semi-automat pentru documente istorice. Arhitectura proiectului cu fluxul de lucru este afișată în Figura 1.7.

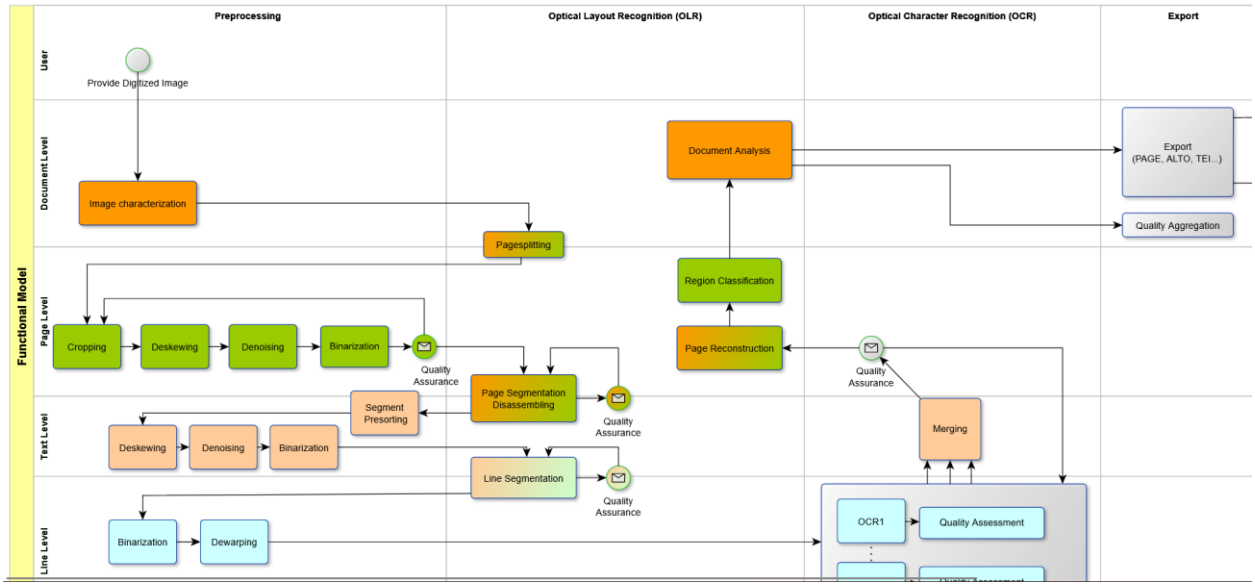



**Figura 1.7. Fragment din structura proiectului *OCR-D*[19].**

[19] https://ocr-d.de/en/about

Multe instrumente propuse în lumea digitizării documentelor istorice sunt specifice și particulare, cum ar fi, de exemplu, cele pentru generarea de seturi de date OCR artificiale pentru limba rusă [79], arabă [80] și română [81]. Cu toate acestea, funcționalitatea lor se limitează doar la unele sarcini, nu și la procesul integral. Luând în considerare acest lucru, putem conclude că valoarea platformelor de digitizare este mult mai mare.

### Metode de postprocesare a documentelor după recunoașterea optică a caracterelor

Postprocesarea OCR sau post-corectarea OCR este o sarcină fie manuală, semiautomată sau automată de verificare și redactare a textului recunoscut de un motor OCR. Postprocesarea integrată într-un sistem OCR oferă o valoare semnificativă rezultatului obținut, astfel încât face sistemul OCR să fie mai robust și mai valoros în ceea ce privește digitizarea în masă a documentelor istorice.

Există multe abordări diferite referitoare la postprocesarea OCR. Unele dintre metodele de bază privesc postprocesarea ca pe o sarcină de corectare a ortografiei textului [82-85]. În lucrarea [53], analizată mai sus, autorii folosesc o metodă numită "secvență la secvență" (*sequence-to-sequence* sau *seq2seq*[20]) bazată pe lucrarea [86]. Aceasta metodă poate folosi un vocabular/dicționar de cuvinte sau lexicon pentru a determina erorile OCR, dar poate funcționa și fără dicționar. Metoda corectează token-urile de la intrare și creează un model de eroare, care este un set de reguli dependente de context adnotate cu ponderi, implementate ca un automat ponderat cu stări finite. După ce se creează modelul de eroare, se mai folosește o căutare în dicționarul de cuvinte pentru a valida sau a elimina sugestiile generate de acest model. Pentru instruirea modelelor de post-corectare OCR, autorii au luat un text recunoscut utilizând mecanismul de votare cu 5 modele mixte [53]. Cele mai bune rezultate după postprocesarea seturilor de date de testare recunoscute au fost obținute cu post-corectarea fără dicționar cu 2,7% CER pentru suedeză, 1,7% CER pentru finlandeză. Post-corectarea fără lexic a reușit să îmbunătățească toate rezultatele, iar postprocesarea cu dicționar nu a produs nicio modificare a rezultatului [53].

Majoritatea metodele de postprocesare OCR au cel puțin doi pași: primul pas ar fi generarea candidaților propuși pentru înlocuirea token-urilor greșite; iar al doilea pas ar fi luarea deciziilor de acceptare a corecturilor propuse la primul pas. Alte abordări ar cuprinde trei pași

[20] ***Seq2seq*** este o familie de abordări de învățare automată utilizate pentru prelucrarea limbajului natural [87]. Această abordare se aplică la probleme precum traducerea automată, parafrazarea textului, alinierea și rezumarea textului.

consecutivi: 1. Mărirea dicționarului de cuvinte unde vocabularul inițial este extins cu cuvinte frecvente din textele recunoscute; 2. Clasarea candidaților unde pentru toate erorile presupuse corecturile sunt calculate folosind o abordare bazată pe reguli analitice, iar candidații propuși spre corectare sunt sortați după cel mai probabil candidat; 3. Luarea deciziei: la final se decide dacă eroarea presupusă este înlocuită cu sugestia de corectare sau dacă este lăsată nemodificată. Modelele din învățare automată sunt utilizate pentru a estima riscul de a înlocui greșit un cuvânt corect. Prin urmare, pasul de decizie ajută la evitarea pașilor de corectare eronată. În acest context autorii lucrării [88] au realizat postprocesarea OCR a ziarelor tipărite în limba germană între 1910 și 1920. Ei folosesc un dicționar și un corpus extern împreună cu distanța Levenshtein și frecvențele pe *n-grame*[21] pentru a vota candidații și a găsi câștigătorul (candidatul cu cele mai multe voturi). Modelul lor funcționează bine atunci când rezultatul corect nu are distanța Levenshtein de la token-ul candidat mai mare decât 2. Un alt exemplu în această direcție este descris în [89], unde documente istorice arabe cu rezultate OCR sub 70% acurateţe la nivel de cuvinte sunt post-corectate și se ajunge la un rezultat de peste 90% acurateţe la nivel de cuvinte. Ei folosesc un dicționar de cuvinte pentru a căuta și identifica cuvintele greșite din documentele recunoscute, pentru care creează candidați de corectură cu un model de regresie. În a doua etapă, candidatul este selectat din nou folosind un model de regresie, dar de data aceasta pe baza caracteristicilor cuvintelor obținute dintr-un model de limbă, construit dintr-un set mare de date text.

Post-corectarea OCR manuală poate da rezultate de înaltă calitate, de multe ori mai bune decât rezultatele automatizate complet, dar necesită timp și efort, uneori și specialiști, când este vorba de documente istorice tipărite cu alfabete vechi care nu se mai folosesc la moment [81]. Oamenii care fac corectarea manuală trebuie să aibă o bună cunoaștere a limbii documentului și să fie instruiți cum să utilizeze instrumente de post-corectare. Există abordări semiautomate care fac ca corectarea manuală să devină ușoară și eficientă. În abordarea prezentată în [90], este propusă o interfață interactivă pentru a corecta rezultatele recunoașterii textului scris de mână. Corecturile utilizatorului sunt luate în considerare în timp real și se folosesc ulterior pentru a oferi sugestii de corectare mai bune.

Un instrument pentru corectarea semi-automată a textului OCR este *PoCoTo* [91]. Instrumentul a fost dezvoltat pentru prima dată ca aplicație desktop și ulterior ca aplicație Web. PoCoTo utilizează un mecanism ce calculează cu o anumită probabilitate care cuvinte dintr-un

[21] ***N-gramele*** sunt secvențe de elemente așa cum apar în texte. Aceste elemente pot fi cuvinte, caractere, sau orice alte elemente pe măsură ce se întâlnesc unul după altul. „N” sau „n” din termenul „n-grame” corespunde numărului de elemente din succesiune [92].

document OCR prezintă erori, folosind o combinație de metode statistice și analitice. Interfața grafică cu utilizatorul îi prezintă utilizatorului fragmente de pagină împreună cu textul OCR care conține posibile erori și oferă, de asemenea, posibile corecții. Astfel, pentru utilizatori este mai ușor să corecteze manual erorile. În anul 2019 a fost publicată o versiune îmbunătățită și complet automatizată a PoCoTo, denumită *A-PoCoTo* [93]. Abordarea OCR a acestui instrument are un model de învățare automată pentru a clasifica candidații care necesită corectare. La intrare, A-PoCoTo primește un set de $n \geq 1$ rezultate OCR paralele pentru un document istoric. Pentru a obține $n > 1$ rezultate OCR paralele pentru un document, pot fi utilizate mai multe motoare OCR. După alinierea preliminară a tuturor rezultatelor OCR disponibile, postprocesarea OCR pentru un document începe cu extinderea dicționarului de cuvinte. Documentele de intrare conțin adesea denumiri speciale (nume de persoane, nume geografice etc) și alte expresii. Orice cuvânt care nu se găsește în dicționar este considerat candidat pentru a extinde dicționarul. În mod intuitiv, dacă există dovezi că un cuvânt din textul recunoscut nu este o eroare de recunoaștere, atunci are sens ca utilizatorul să-l includă în vocabular. Acest lucru implică faptul că acest cuvânt poate servi drept candidat pentru corecțiile ulterioare.

În lucrarea [94], se prezintă o metodă interesantă care se bazează pe faptul că erorile OCR pentru același cuvânt, datorate asemănării semantice, sunt grupate în spațiul vectorial. Autorii utilizează un model *Word2Vec*[22] pentru a obține grupuri de erori OCR și sinonime ale cuvintelor. Apoi, prin verificarea cuvintelor corecte folosind un dicționar, grupurile de cuvinte greșite și cele corecte sunt identificate folosind distanța *Levenshtein*. Prin utilizarea acestui set de date paralel, un model neural de traducere automată este antrenat, efectuând o traducere a cuvintelor greșite în cuvinte corecte.

O abordare similară cu cea din [94] este prezentată în [95], unde autorii propun un instrument de postprocesare OCR care grupează, de asemenea, toate cuvintele similare din text, dar în loc să fie grupate semantic, ele sunt grupate după distanțele de caractere în spațiul euclidian. Ei folosesc date externe precum lexicoane și reguli morfologice pentru a propune candidați la corectare. În cele din urmă, corectarea se efectuează prin selectarea candidatului cu cel mai mare punctaj.

[22] ***Word2vec*** este o familie de arhitecturi de modele și optimizări care pot fi folosite pentru a învăța caracteristici de cuvinte din seturi mari de date [96]. Caracteristicile învățate prin *word2vec* s-au dovedit a fi de succes într-o varietate de sarcini de procesare a limbajului natural din aval.

**Instrumente de procesare a textelor istorice tipărite in limba română**

În acest compartiment vom descrie unele lucrări din domeniul procesării textelor istorice tipărite în limba română.

Un proiect important în acest domeniu este ***DeLORo***[23]. Obiectivele acestui proiect îl constituie dezvoltarea unei tehnologii capabile să descifreze documente scrise în limba română cu caractere chirilice și să le translitereze în caractere latine, fundamentând astfel posibilități de studiere și conservare a tezaurului cultural [97, 98]. În cadrul proiectului autorii au reușit să dezvolte un instrument online de adnotare a imaginilor din documente chirilice românești. Tot aici au fost puse bazele elaborării unei resurse importante și anume corpusul chirilic românesc vechi *ROCC* [98], care include o colecție de documente istorice scanate, adnotate cu text transcris [99]. Corpusul constă din 367 de pagini de document scanate, cu un total de 6418 linii de text adnotate.

Detaliile privind resursele și tehnologia elaborate în proiectul *DeLORo* sunt descrise în Cristea et al. [98]. În această lucrare se propun soluții pentru crearea setului de date în procesul de antrenare al rețelelor neurale, dar și folosirea acestora la antrenarea rețelelor neurale propriu-zise pentru sarcini precum segmentarea și OCR. În această activitate autorii construiesc corpusul *ROCC*. Colecția de documente din ROCC acoperă secolele XVI-XIX, operând cu diferite nivele de calitate ale documentelor istorice. Aceste documente sunt organizate în 3 nivele de dificultate, 3 tipuri de scriere și 3 nivele de adnotare. Astfel, corpusul *ROCC* posedă următoarele nivelele de dificultate: I – documente ușor de procesat, pagini relativ curate, linii aliniate, fonturi obișnuite; II – dificultate medie de prelucrare, existența petelor, unele anomalii de font, linii nealiniate, scrieri între rândurile de text (ligaturi); și III – documente dificil de prelucrat, pagini cu multe pete, fonturi foarte neuniforme, linii cu curburi, scrieri frecvente între rândurile de text. Tipurile de scriere propuse sunt: p – tipărit, u – uncial, m – manuscris cu ligaturi; iar nivele de adnotare sunt: o – original neadnotat; g – adnotat de experți umani, cu imagini aliniate manual cu transcripții; t – aliniat și interpretat automat. Procesul de colectare a documentelor în acest corpus este iterativ și include imagini originale ale paginilor în grafia chirilică, adnotări manuale referitoare la *segmentarea coloanelor, rândurilor de text, cuvintelor, caracterelor*. Adnotarea manuală se referă la două componente: segmente vizuale în *coloane, rânduri, scrierea dintre rânduri* sau *marginală, cuvinte, caractere*; și *transcrierile lor în grafie latină*. La elaborarea resurselor necesare pentru antrenarea rețelelor neurale, autorii se axează pe două tipuri din ele: un model de limbă care să conțină cât mai multe forme de cuvinte din cele scrise vreodată în limba română în grafia chirilică

[23] The DeLORo project PN-III-P2-2.1-PED-2019-3952, no. 400PED: "Deep Learning for Old Romanian" (http://deloro.iit.academiaromana-is.ro/).

și o colecție mare de grafeme identificate și transliterate. Colectarea primului tip de resurse este foarte dificilă din mai multe motive. Un motiv provine din faptul că în multe perioade istorice nu au existat norme de utilizare a limbii, astfel încât există o mare diversitate de forme scrise ale cuvintelor pe dimensiunea temporală și spațială. Nu au fost identificate modele de flexionare pentru româna veche, care ar fi permis o generare automată a vechiului lexic românesc; substantivele proprii sunt aproape imposibil de inventariat în totalitate. Acest tip de resurse acționează ca un vocabular, iar autorii vin să propună aplicarea metodelor de augmentare automată a acestora. Al doilea tip de resurse din corpusul ROCC, sunt utilizate pentru instruirea rețelelor neurale pe diferite calități de documente și diferite tipuri de scris. Resursele ar trebui să ajute la sarcina de clasificare, unde se identifică și se etichetează obiectele vizuale care apar pe o pagină scanată. Etichetele propuse sunt *coloană, rând în coloană, cuvânt în linie, cuvânt pe margine, literă în cuvânt, literă pe margine, accent deasupra literei etc.* precum și *literele alfabetului latin cu diacritice specifice românești.* Obiectele sunt încadrate pe forme de delimitare înconjurătoare având forma dreptunghiulară, fiecare caracterizată de 4 coordonate: *<x1, y1>* – coordonatele colțului din stânga sus și *<x2, y2>* – din colțul din dreapta jos. În Figura 1.8, autorii evidențiază unele obiecte vizuale pe un fragment de pagină. Etichetarea obiectelor vizuale se face de către lingviști, inclusiv doctoranzi și masteranzi în Lingvistică prin instrumentul front-end *OOCIAT*.

Fără îndoială, la antrenarea rețelelor neurale sunt necesare seturi mari de date de antrenare. La etapa OCR autorii învață rețele neurale cu perechi de semne chirilice grafice românești și transcripțiile lor corespunzătoare în latină. Pe lângă adnotările efectuate prin interfața *OOCIAT*, autorii folosesc și resurse alternative din *Monumenta Linguae Dacoromanorum (MLD)* [100] și corpusul *UAIC-RoDia Treebank* [101] – o colecție de arbori sintactici de propoziții din limba română veche. Volumul setului de date încă este insuficient. O metodă de a augmenta setul de date propus, este utilizarea iterativă a ceea ce a fost adnotat manual pentru etapa OCR, urmată de corectarea manuală a rezultatele OCR. Autorii consideră că acest proces poate accelera adnotarea manuală, deoarece, pentru următorul grup de pagini noi, precizia modulelor OCR ar fi mai mare și, în mod corespunzător, efortul manual de corecție ar trebui să scadă. O altă metodă propusă, este de a folosi un document transcris manual și de a alinia imaginile paginilor cu textul corespunzător, rând cu rând.

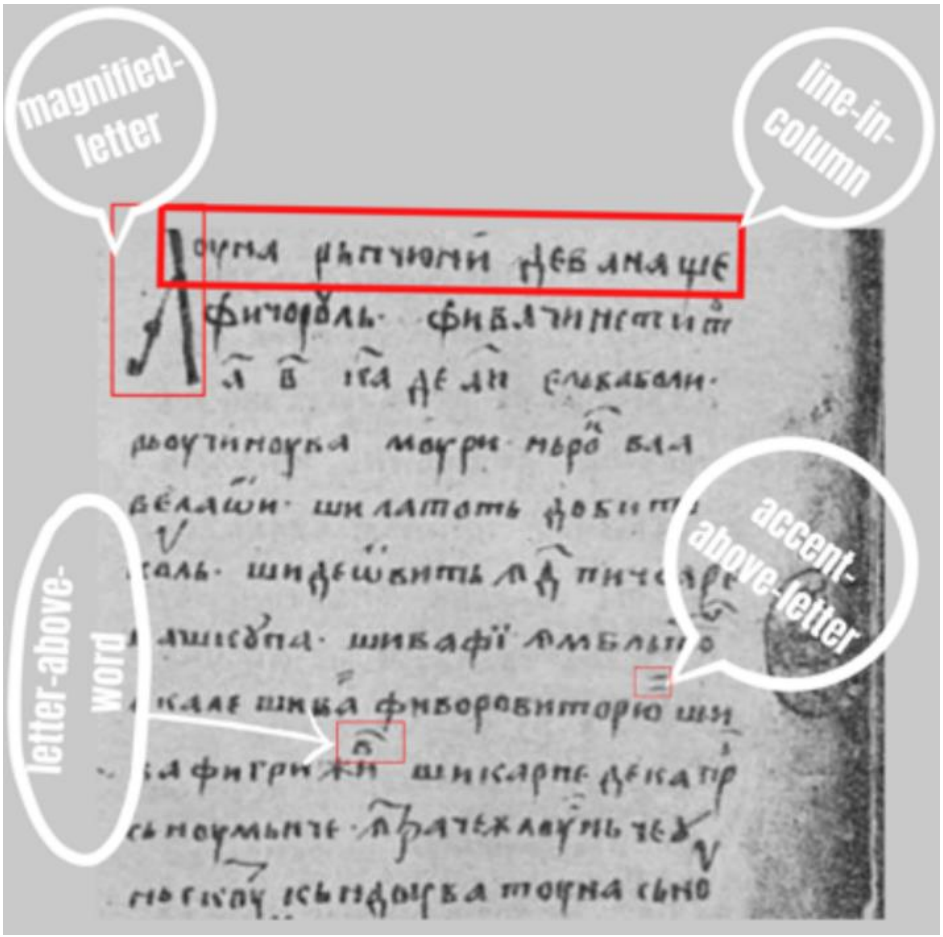



**Figura 1.8. O ierarhie de obiecte adnotate pe o pagină scanată care conține scriere uncială [98].**

Procesul de aliniere presupune segmentarea imaginii fiecărei pagini în rânduri de text și fiecare rând de text – în caractere, apoi aplicarea modelului OCR pe fiecare rând identificat și recunoașterea caracterelor. Aplicând această strategie asupra unui document, în mod repetat, modulul OCR este îmbunătățit constant. În experimentele de antrenare a modelului OCR au fost folosite o combinație între un model statistic pentru extragerea de caracteristici și o rețea neurală cu arhitectura CNN, menită să descopere obiecte și să le atribuie etichete. Pentru antrenarea rețelei sunt utilizate imaginile adnotate manual din ROCC și documentele aliniate. Modelele OCR pentru recunoașterea obiectelor din colecțiile tipărite au rezultate foarte bune, dar nu și pentru colecțiile de manuscrise, o sarcină încă neabordată de autori. Detectarea obiectelor de tip linie și caractere este realizată de un algoritm *Faster R-CNN* compus din 3 rețele neuronale reziduale [102]. Rezultatul segmentării liniilor pentru o pagină de text este prezentat în Figura 1.9. După segmentarea rândurilor, același algoritm este utilizat pentru a extrage fiecare literă, care apare în acest rând pentru a o recunoaște direct în calitate de caracter din alfabetul latin, prin omiterea

pasului intermediar de recunoaștere în grafia chirilică. Autorii estimează succesul acestei tehnologii în funcție de o combinație de criterii temporale, de dificultate și de scriere, după cum urmează: pentru fiecare dintre 7 perioade istorice de 50 de ani, de la începutul secolului al XVI-lea până la mijlocul secolului al XIX-lea se ia în considerare un eșantion aleatoriu de 45 de pagini. Un eșantion este considerat pozitiv dacă 4 din cele 5 pagini trec testul, iar o pagină de test este promovată dacă rata de eroare de recunoaștere a caracterelor izolate de pe pagina respectivă este foarte mică. Din totalul de 315 mostre, autorii doresc să obțină o rată minimă de 80% de teste pozitive, adică minim 252 de pagini.

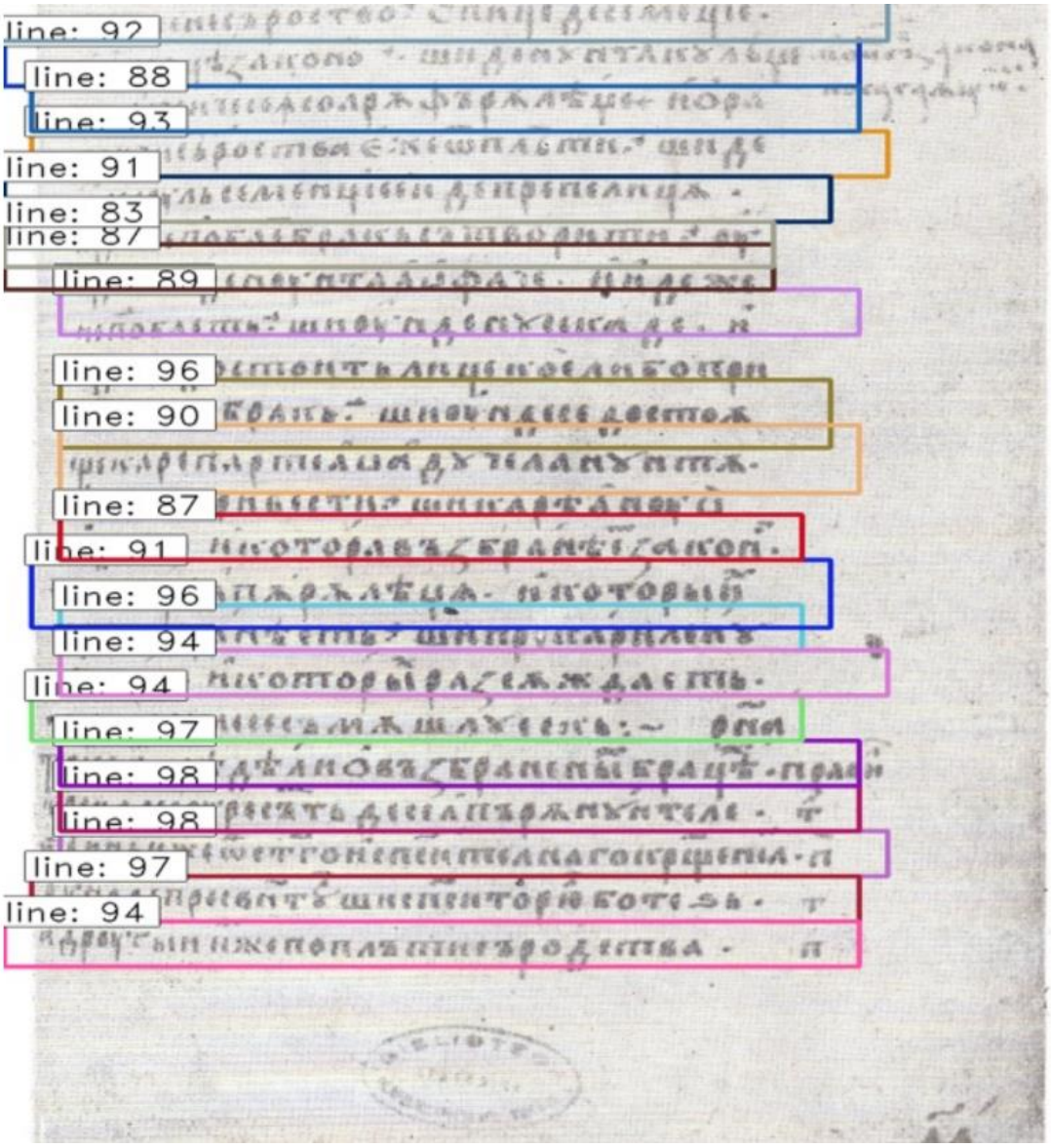



**Figura 1.9. Exemplu de dreptunghiuri de delimitare prezise [98].**

Luând în considerare că rezultatele obținute în final sunt secvențe de caractere delimitate doar de rând nou, iar pe lângă acest fapt în tipăriturile chirilice vechi cuvintele sunt rareori separate

prin spații care se pot distinge, autorii propun în continuare și metode de separare a cuvintelor. Presupunând că fiecare caracter din imaginea originală a unui rând de text este recunoscut, rămâne încă problema segmentării șirului de litere latine românești în cuvinte. Pentru a face acest lucru, se folosește o abordare secvență la secvență, care primește la intrare o secvență de litere și, pentru fiecare literă aparte, recunoaște una dintre cele 4 clase: *începutul cuvântului, sfârșitul cuvântului, mijlocul cuvântului și cuvântul cu un singur caracter*. Autorii nu folosesc niciun context pentru a schimba sensul unei litere chirilice ambigue (exemplele sunt: *Ѧ ѧ: ĭa/ ea; Ѳ ѳ: th, ft*). O altă acțiune implementată de către autori este clusterizarea lexicală, pentru a grupa forme vechi de cuvinte aparținând aceleiași leme și aceleiași părți de vorbire. Pentru aceasta, autorii intenționează să aplice *string kernels* [103] și *clusterizarea spectrală* [104]. Într-o primă etapă formele flexionate ale aceleiași leme, găsite într-o colecție de documente vechi românești chirilice, vor fi etichetate ca aparținând aceleiași clase. Într-o a doua etapă, clusterele detectate ar trebui să fie aliniate cu intrările de dicționar ale unui dicționar al limbii române moderne. În continuare lucrărilor sale, autorii planifică integrarea instrumentelor elaborate într-o platformă cu acces gratuit pentru cercetători.

În lucrarea M. Găman et al. [105], autorii propun o abordare specială pentru detectarea rândurilor de text din documente vechi românești. Tehnologia se bazează pe învățarea aprofundată cu *auto-ritm*[24] (self-paced) [106-108], capabilă să îmbunătățească performanța de detectare a rândurilor de text din documentele vechi. Folosind o metodă specială de învățare cu auto-ritm, autorii antrenează un model de detectare a rândurilor de text într-un număr *k* de iterații. La fiecare iterație, se combină dreptunghiurile de delimitare a adevărului de bază cu dreptunghiurile de delimitare prezise de model, iar adnotările noi obținute se includ la următoarea iterație de antrenare. Autorii fac experimente de detectare a rândurilor de text pe două seturi de date cu documente istorice din corpusurile *ROCC* și *cBAD* [109] – un corpus format din 2035 de pagini de documente istorice scrise în grafie latină, comparând modelul *YOLOv4* [110] cu o versiune de *YOLOv4* antrenată prin abordarea de învățare cu auto-ritm. Aceștia demonstrează că strategia de învățare cu auto-ritm aduce câștiguri semnificative de performanță, îmbunătățind precizia medie a YOLOv4 cu mai mult de 12% pe set de date din ROCC și 39% pe setul de date din cBAD (vezi tabelul 1.4).

[24] Oamenii au capacitatea de a se ghida în procesul de învățare, fiind capabili să învețe concepte noi în ritmul lor liber, fără a necesita instrucțiuni de la un profesor. De exemplu, majoritatea studenților aleg, când și cât timp să studieze, folosind un proces de învățare a curriculum-ului într-un ritm propriu sau auto-ritm [105].

**Tabelul 1.4. Rezultatele pentru YOLOv4 (de bază) versus YOLOv4 bazat pe învățarea cu auto-ritm. Scorurile pentru precizia medie (AP) sunt raportate pe două seturi de date: ROCC și cBAD. Cele mai bune scoruri pentru fiecare set de date sunt evidențiate cu caractere aldine [105].**

| Data Set | Model | Iterație | AP (%) |
|---|---|---|---|
| ROCC | YOLOv4 (de bază) | - | 81.55 |
| | YOLOv4 + auto-ritm | 1 | 72.43 |
| | | 2 | 87.22 |
| | | 3 | 88.05 |
| | | 4 | 89.86 |
| | | (final) 5 | **93.73** |
| cBAD | YOLOv4 (de bază) | - | 35.37 |
| | YOLOv4 + auto-ritm | 1 | 22.52 |
| | | 2 | 54.81 |
| | | 3 | 63.14 |
| | | 4 | 63.72 |
| | | (final) 5 | **74.57** |

Luând în considerare rezultatele prezentate în tabelul 4, autorii ajung la concluzia că metoda de învățare cu auto-ritm este extrem de utilă în îmbunătățirea detectării rândurilor de text cu etichete lipsă și își propun să extindă aplicabilitatea metodei de învățare cu auto-ritm elaborată de ei la alte sarcini de detectare care au problema etichetelor lipsă.

Vom menționa, de asemenea, contribuțiile aduse la subiectul de valorificare a patrimoniului constituit din tipărituri vechi și contemporane în lucrarea D. Gîfu [111], unde autoarea propune dezvoltarea unui lexicon din ziare românești începând cu secolul al XVIII-lea, care acoperă trei regiuni românești: Moldova, Transilvania și Țara Românească pentru analiza diacronică a cuvintelor vechi extrase din colecția de ziare pe baza anului documentelor în care se regăsesc aceste cuvinte. Lexiconul obținut din intervalul anilor 1829-2015 include: din Moldova – 65901 de cuvinte dintre care 5085 de cuvinte vechi; din Țara Românească – 137261 (6525 cuvinte vechi); din Transilvania – 160923 (21023 cuvinte vechi); de asemenea sunt incluse și cuvintele din texte tipărite în Basarabia – 107324 (703 cuvinte vechi). Această colecție de texte a fost adunată și prelucrată automat cu instrumente de procesare a limbajului natural, precum *segmentarea*, *tokenizarea*, *lematizarea*, *analizarea morfo-sintactică*, *recunoașterea entităților de nume* și altele. În această colecție de articole un număr important de cuvinte vechi sunt extrase și etichetate cu două clase *an* și *regiune*. Adnotarea automată a cuvintelor s-a bazat pe metodologia de căutare a cuvintelor într-un dicționar electronic și anume *DEX-online*[25]. Astfel, cuvintele negăsite (necunoscute) sunt adnotate cu o etichetă specială și analizate ulterior din perspectiva diacronică utilizând *Dicționarul Tezaur al Limbii Române* în versiune electronică (*eDTLR* [112]).

[25] https://dexonline.ro/

Acest proiect vine în sprijinul lexicografilor, antropologilor, jurnaliștilor, dar și în ajutorul cercetătorilor preocupați de digitizarea documentelor vechi românești reprezentând o resursă lingvistică prețioasă folosită în instrumentele OCR și de transliterare.

Aici vom finaliza analiza instrumentelor și metodelor de digitizare și procesare a documentelor istorice. În continuare vom analiza documentele istorice din patrimoniul românesc.

### 1.3. Documente istorice românești tipărite cu alfabete chirilice

Pentru a demonstra necesitatea instrumentelor de digitizare și procesare a documentelor românești tipărite cu alfabete chirilice vom evidenția evoluția, numărul estimativ și diversitatea documentelor românești tipărite cu alfabete chirilice.

Limba română a parcurs o cale lungă de dezvoltare, iar în evoluția limbii române literare se disting două epoci fundamentale: epoca veche și epoca modernă, fiecăreia fiindu-i subsumate perioade ale căror limite nu sunt întotdeauna indicate. Autorii lucrării [113] afirmă că prima perioadă a epocii vechi „durează până în mijlocul veacului al XVII-lea”, deci până în 1650. „Între anii 1640-1650 apar *Cazaniile* de la Bălgrad (1641), *Cazania* lui Varlaam (1643), *Pravila* de la Govora (1640), *Pravila* lui Vasile Lupu (1646), *Noul Testament* de la Bălgrad (1648)”. Prima lucrare imprimată pe teritoriul românesc - la Târgoviște - a fost Liturghierul slavon, de ieromonahul Macarie în 1508, a cărei exemplar facsimil se regăsește și în Biblioteca Națională a Republicii Moldova, în timp ce prima publicație tipărită în română a fost cartea „Întrebare creștinească” sau „Catehismul” (diaconul Coresi, Brașov, 1535) [114].

La momentul actual, majoritatea colecțiilor tipărite cu alfabete chirilice se păstrează în biblioteci din Republica Moldova și din România, precum Biblioteca Națională a Moldovei, Biblioteca Națională a României, Biblioteca Științifică Centrală „A. Lupan”, Biblioteca Academiei Române, Biblioteca Centrală a Universității de Stat din Moldova etc. Documente românești tipărite cu alfabete chirilice pot fi găsite și în bibliotecile altor țări, în mod special, mai multe din acestea se regăsesc în bibliotecile și arhivele din Sankt Petersburg.

Biblioteca Națională a Republicii Moldova deține o colecție de aproximativ 21000 cărți vechi și rare. Circa 20 de cărți din această colecție sunt tipărite în limba română, la Chișinău și Dubăsari, utilizându-se alfabetul chirilic și alfabete de tranziție [115]. Bibliotecile din Sankt Petersburg găzduiesc valoroase exemplare de tipărituri românești din perioada secolelor 17-19. De exemplu, din cele 66 de titluri prezentate în „*Catalogul ediţiilor chirilice ale slavilor de sud şi ale românilor*”, 45 aparțin slavilor de sud, iar restul de 21 sunt atribuite țărilor românești [116].

O sursă importantă cu colecții scanate din secolul XX este Biblioteca Națională Digitală - *Moldavica*[26], care conține o colecție de documente patrimoniale incluse în Programul Național "Memoria Moldovei". În *Moldavica* sunt plasate colecțiile de cărți vechi tipărite cu caractere slavone și caractere moderne (alfabet de tranziție, chirilic românesc, latin), care s-au păstrat până în prezent în exemplare numărate în Moldova, atingând cifra de 1194 de documente scanate, iar colecția de reviste naționale de la 1867 până la 1945 conține 4276 documente etc.

Bibliografia românească veche din anii 1508-1830[27] a Bibliotecii Academiei Române numără peste 1960 de tipărituri, fără a se lua în considerare periodicele. Majoritatea documentelor (79%) sunt tipărite cu alfabete chirilice în limba română. De asemenea, aici sunt și tipărituri în limba greacă, slavonă, franceză, engleză, bulgară, arabă, germană etc (vezi Figura 1.10). Dacă să vorbim despre tipărituri scanate, atunci Biblioteca Digitală Națională[28] a Bibliotecii Naționale a României include 724 periodice românești vechi și 1228 de resurse de carte românească veche și bibliofilă;

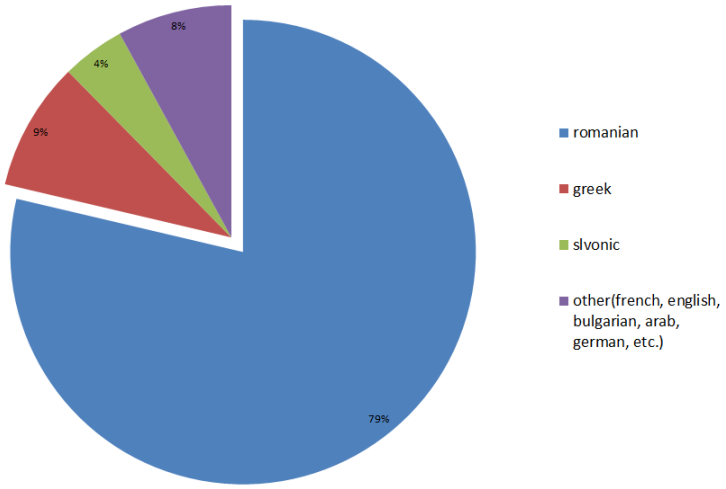



**Figura 1.10. Statistica tipăriturilor din anii 1508-1830 de la Bibliotecii Academiei Române**

Tipografia din Țara Românească a apărut printre primele din Europa. Aceasta își are începutul în secolul al XVI-lea, pe parcursul căruia s-a tipărit cu litere chirilice ale alfabetului chirilic român. Un rol special l-a avut ieromonahul Macarie, venit în Țara Românească la începutul secolului al XVI-lea, cu meseria de tipograf învățată la Veneția, și având experiența de tipograf al voievodului Muntenegrului, unde între anii 1493-1496 a tipărit șase cărți [115]. Ieromonahul Macarie a tipărit pe pământul românesc trei cărți slavone de factură religioasă: un *Liturghier* în

[26] http://www.moldavica.bnrm.md/
[27] https://biblacad.ro/bnr/brv.php
[28] http://digitool.bibnat.ro/R

1508, un *Octoih* în 1510 și un *Evangheliar* în 1512. În ceea ce privește locul instalării tiparului în Țara Românească, se presupune că ar fi mănăstirea Dealu, de lângă Târgoviște și Mănăstirea Vâlcea, unde au fost identificate cele mai multe exemplare de cărți macariene [115]. În activitatea tipografică din Țările Române în timpul lui Matei Basarab și a lui Vasile Lupu în Moldova a fost tipărită *Îndreptarea Legii* în anul 1652 în tipografia de la Târgoviște, tradusă din greacă, de care s-au folosit românii din Țara Românească, Moldova și Transilvania timp de secole [115]. Unele calcule aproximative despre numărul titlurilor de carte apărute între anii 1653-1656 arată că producția de carte din această perioadă crește față de perioadele anterioare. Potrivit acestora, în tipografiile românești au apărut 43 de cărți, 23 în limba română, 13 în limba slavonă, 5 slavo-română și 2 în limba greacă [115]. Până în anul 1830, tipografiile au avut o creștere aproape exponențială a tipăriturilor.

Activitatea tipografică românească în perioada 1508-1830 a fost destul de fructuoasă. Cartea tipărită în Țara Românească a fost difuzată în toate provinciile românești, fiind editată uneori în tiraje foarte mari, unele dintre care s-au păstrat și astăzi, urmând a fi digitizate și valorificate în tezaurul nostru românesc.

### 1.4. Concluzii la capitolul 1

După analizarea instrumentelor și metodelor de digitizare a documentelor istorice concluzionăm că există, dar și se mai dezvoltă o mulțime de metode, instrumente, resurse și chiar platforme care oferă posibilitatea de a preprocesa, recunoaște, postprocesa, translitera documentele istorice într-un mod rapid și eficient. Mai mult ca atât, metodele noi de recunoaștere bazate pe antrenarea modelelor OCR pe imagini cu linii de text [53] sporesc viteza de antrenare a motoarelor OCR, iar prin urmare – și viteza de digitizare în masă. După recunoaștere, instrumentele de postprocesare OCR asigură obținerea unui text de calitate, care poate fi utilizat în procesări lingvistice ulterioare, la plasarea în web pentru studiere, la extinderea metadatelor din bibliotecile digitale etc. Un aport valoros îl constituie dezvoltarea platformelor de digitizare, care integrează majoritatea sarcinilor de procesare cu succes a documentelor istorice. Un efort semnificativ pentru decodificarea documentelor vechi românești este efectuat în proiectul *DeLORo* [98], unde autorii au venit cu unele soluții promițătoare, care își aduc contribuția la păstrarea și asigurarea accesului la textele din tezaurul chirilic românesc.

Cu toate acestea, nu există un instrument complet funcțional, adaptat anume pentru digitizarea și transliterarea textelor vechi românești, care ar include toate etapele indispensabile procesului (preprocesare, recunoaștere și transliterare, post-procesare). Necesitatea instrumentelor

de digitizare și procesare a documentelor românești tipărite cu alfabete chirilice este argumentată și prin numărul și diversitatea documentelor românești tipărite cu alfabete chirilice, care la momentul actual se află în bibliotecile din Republica Moldova și din România, dar și în bibliotecile din alte țări.

Luând în considerare aceste date putem spune cu siguranță că există un teren vast de activități privind digitizarea și procesarea documentelor istorice românești, iar cercetarea și dezvoltarea instrumentelor de procesare computerizată a acestora este o prioritate pentru Moldova și România.

# 2. TEHNOLOGII DE PROCESARE A TEXTELOR ROMÂNEȘTI DIN SEC. XVII-XX

În acest capitol vom fundamenta abordările noastre în proiectarea tehnologiei de procesare a textelor istorice (tipărite în limba română cu caractere chirilice începând cu secolul XVII), descriind metodele elaborate și argumentând utilizarea anumitor module din categoria celor existente. Din ele fac parte: tehnologia de *recunoaștere optică a caracterelor* (OCR) care la rândul său include: *preprocesarea imaginilor*, *analiza și segmentarea machetei documentului* și *clasificarea fonturilor*; *tehnologia de transliterare din alfabetul chirilic românesc în alfabetul modern*; *instrumentarul de aliniere a textelor vechi la texte moderne.* Vom începe prin a descrie setul de acțiuni întreprinse la recunoașterea textelor din secolul XVII. Vom menționa, că procesul este organizat pe principiul utilizării tehnologiilor convergente, adică cel de interconectare în cadrul unei platforme a aplicațiilor din anumite domenii, de rând cu elaborarea componentelor proprii.

## 2.1. Descrierea procesului de recunoaștere optică a caracterelor

Analiza sistemelor de recunoaștere optică a caracterelor, efectuată în capitolul precedent, ne-a oferit argumente în favoarea utilizării în scopul recunoașterii caracterelor chirilice românești a programului ABBYY FineReader Professional. Primele testări și adaptări le-am efectuat pe versiunea FR 12, optând pe parcurs pentru versiuni mai noi, precum FineReader 14 și FineReader 15. Am constatat, că atât FR12, cât și versiunile ulterioare, nu sunt orientate apriori spre procesarea textelor vechi românești, astfel eforturile noastre s-au concentrat pe extinderea capacităților acestui produs pentru adaptarea lui la soluționarea problemelor menționate mai sus. În scopul operării cu documente din perioada indicată a fost necesar de elaborat un șir de componente noi (în special, alfabete și dicționare), precum și de antrenat acest soft pe seturi de date adiționale, asigurându-se un grad pe cât e posibil de înalt al calității rezultatului. În finalul acestor acțiuni se creează unul sau mai multe modele orientate la recunoașterea textelor dintr-o anumită perioadă istorică.

FineReader este un program informatic OCR dotat cu funcțiile necesare preprocesării și recunoașterii documentelor, utilizarea căruia contribuie semnificativ la creșterii productivității executării acestor operațiuni. Acesta oferă unelte de înaltă performanță, care totodată sunt ușor de utilizat pentru accesarea informațiilor cuprinse în documentele tipărite și în fișierele PDF. În Figura 2.1, prezentăm un document din secolul XVII procesat cu FR 12. Procesarea include aplicarea rezoluției optimale (800-1200 dpi în cazul de față), ștergerea manuală a petelor de uzură din imagine, antrenarea modelului OCR și recunoașterea tuturor paginilor documentului, dar și

corectarea manuală a rezultatelor OCR. Toate aceste operațiuni pot fi executate din fereastra FR 12 prezentată în Figura 2.1. Ținem să menționăm că unele operații de procesare a imaginii, precum binarizarea și îndreptarea rândurilor s-a efectuat cu alt instrument software și anume cu Scan Tailor. Despre acest instrument vom discuta în următoarea secțiune.

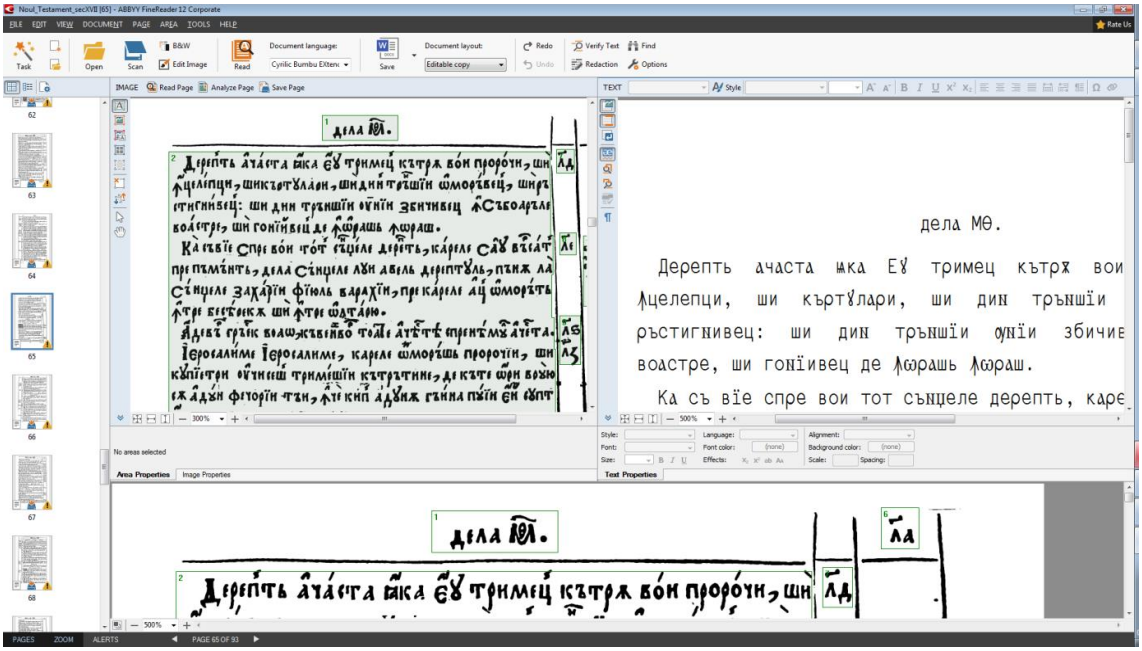



**Figura 2.1. Fereastra principală din FR 12 reprezentând interfața grafică cu utilizatorul.**

Procesul de recunoaștere optică a caracterelor aplicat pe un document, care a fost tipărit într-o limbă ce nu se conține în FR (pentru softul respectiv aceasta fiind o limbă nouă - limba utilizatorului), constă din câteva etape. Cele mai importante etape, pe care le parcurge un document în curs de recunoaștere optică a caracterelor, sunt următoarele:

1. *Preprocesarea imaginii*. Această etapă presupune în cele mai multe cazuri editarea imaginii, curățarea de elemente irelevante sau conținutul zgomotos de pe imagine, îndreptarea rândurilor și ajustarea rezoluției la un nivel optimal. În cazul nostru realizarea acestor acțiuni o putem efectua cu softuri existente, dar găsind un mod specific de aplicare pentru textele vechi.
2. *Pregătirea și crearea limbii și a alfabetului.* La această etapă se pregătesc toate caracterele componente (litere, semne de punctuație, ligaturi – prin ultima noțiune având-se în vedere reunirea mai multor litere într-un singur semn grafic) ale limbii în cauză. Dacă alfabetul conține caractere din mai multe limbi, mai întâi se aleg limbile de bază, după care din fiecare limbă se aleg toate caracterele de care este nevoie pentru recunoaștere.
3. *Pregătirea și crearea dicționarului de cuvinte.* Orice limbă (ca instrument) integrată în FR12 are vocabularul său. Dacă creăm un limbaj nou, adică cel al utilizatorului, este nevoie

de creat un vocabular specific acestei limbi. De exemplu, pentru documentele în limba română tipărite cu litere chirilice românești, avem nevoie de cuvinte românești în grafie chirilică, astfel, o soluție ar fi transliterarea cuvintelor din grafia modernă (latină) în cea chirilică, și adăugarea acestora în dicționarul din FR 12.

4. *Crearea și antrenarea șabloanelor.* Fiecare caracter în parte este învățat de mașină într-un mod supervizat (adică cu implicarea specialiștilor), astfel încât caracterelor segmentate din imagine li se pun în corespondență caracterele digitale Unicode. În urma antrenării motorului OCR din FR, de regulă - pe un volum de 2-10 pagini ale documentului cercetat, se creează un șablon (model) caracteristic documentului dat, sau, în general, caracteristic tipografiei în care acel document a fost tipărit. Numărul de pagini, necesare pentru obținerea unui rezultat de calitate depinde de specificul documentului. În procesul de învățare FR utilizează nu doar rezultatele stabilite în urma procesării documentului respectiv, dar și o serie de modele de rețele neurale antrenate în prealabil (preantrenate), precum și mecanismele incorporate bazate pe analiza statistică, îmbunătățindu-se astfel procesul de recunoaștere. Informațiile de context joacă un rol semnificativ și sunt utilizate în motorul OCR într-o manieră similară citirii textului de către om, cuvintele fiind adesea prezise doar după câteva caractere, ținându-se cont și de context, adică de sensul întregii propoziții. Cu toate acestea, experiența obținută nu este împărtășită între documente diferite, acesta fiind un dezavantaj temporar. În următoarele versiuni ale FR s-ar putea să apară unele arhitecturi avansate de rețele neurale, precum arhitecturi din clasa rețelelor neurale recurente, care vor putea lega experiența (memoria) recunoașterii între pagini sau chiar între documente.

Astfel, din pașii enumerați mai sus, doar pentru primul – cel de preprocesare a imaginii – vom folosi practic fără modificări softuri existente, pentru realizarea celorlalți pași sunt necesare elaborări care extind capacitățile prestabilite ale softului existent. În compartimentul ce urmează vom descrie proprietățile și funcționalitatea modulelor de preprocesare a imaginilor, evidențiind modul specific de aplicare a acestora în cazul operării cu texte vechi.

## 2.2. Preprocesarea imaginilor vechi

În acest compartiment vom descrie particularitățile aplicării a două instrumente pentru procesarea imaginilor – cel incorporat în FR și Scan Tailor[29], elucidând specificul aplicării acestora la procesarea textelor vechi.

[29] https://scantailor.org/

Este cunoscut faptul, că calitatea imaginii influențează semnificativ acuratețea recunoașterii optice a caracterelor, astfel preprocesarea imaginilor din documentele supuse recunoașterii, în special ale celor vechi, în scopul îmbunătățirii aspectului calitativ al acestora, devine o etapă iminentă în algoritmul de lucru. De regulă, timpul își lasă amprenta asupra documentelor istorice, diminuând calitatea acestora. De asemenea, preprocesarea documentelor scanate este un pas important în învățarea automată, deoarece la această etapă datele inițiale sunt adaptate pentru a constitui o intrare compatibilă într-o rețea neurală. În majoritatea cazurilor, rețeaua neurală, care va procesa imaginile, operează cu  o dimensiune de intrare fixă, adică nu poate primi imagini de dimensiuni diferite ca intrare. Astfel, la etapa curentă, documentele colectate ar trebui redimensionate la parametrii specifici. O altă procedură de bază, care se aplică documentelor scanate la etapa de preprocesare, este orientarea corectă a acestora, fapt care va permite afișarea textului într-un mod ușor de citit, de exemplu, din stânga spre dreapta în limba română. Dacă o rețea neurală, care recunoaște caracterele din imagine, nu este învățată să le recunoască în toate pozițiile posibile, atunci orientarea textului din document este esențială.

Astfel, concludem că este necesar să efectuăm mai multe ajustări ale imaginii, înainte ca aceasta să fie supusă recunoașterii. După cum  este ilustrat în Figura 2.3, FR dispune de propriul editor de imagini incorporat. Opțiunile de procesare a imaginii sunt afișate în Figura 2.2.

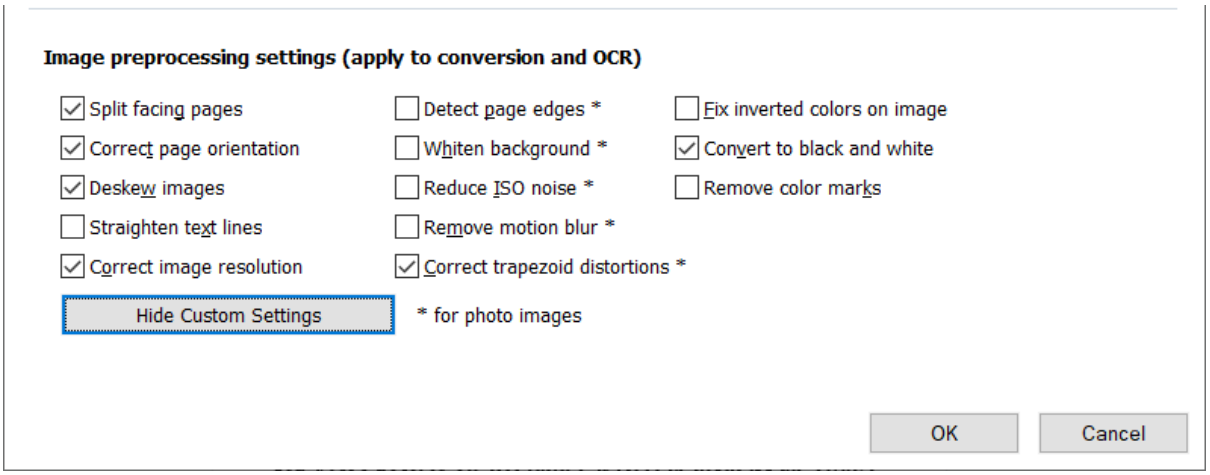



**Figura 2.2. Setările de procesare a imaginii in FR 15.**

Convertirea documentului în alb-negru este posibilă din FR ca opțiune implicită atunci când se configurează setările de *procesare a imaginii (Image processing settings)*. Prin urmare, putem conclude că FR asigură unele opțiuni necesare pentru preprocesarea textelor vechi, însă nu oferă tot spectrul util, fiind necesară implicarea unor instrumente adiționale. Gama acestora, inclusiv cu distribuire gratuită, este destul de amplă. Unul dintre modulele esențiale de preprocesare a documentelor vechi care lipsesc în FR se referă la îngroșarea caracterelor.

Asemenea ruginii care mănâncă fierul, uzura specifică cărților subțiază caracterele, de asemenea unele metode de binarizare a imaginii pot subția liniile din glife, iar pentru a le îngroșa din nou în etapa de preprocesare a imaginii folosim un modul special din Scan Tailor, instrument pe care îl descriem în continuare.

### Preprocesarea cu Scan Tailor

Vom examina, de asemenea, și o altă soluție locală de preprocesare a imaginilor numită *Scan Tailor*. S-a dovedit că aceasta este destul de potrivită și pentru cazul nostru, luându-se în considerare mai mulți factori, printre care sunt: *viteza de procesare; ordinea implicită de procesare a documentului, accesibilă pentru marea majoritate de utilizatori.* Astfel, în cele ce urmează ne vom baza pe aplicarea softului Scan Tailor. Vom examina pașii de procesare a imaginii cu acest soft.

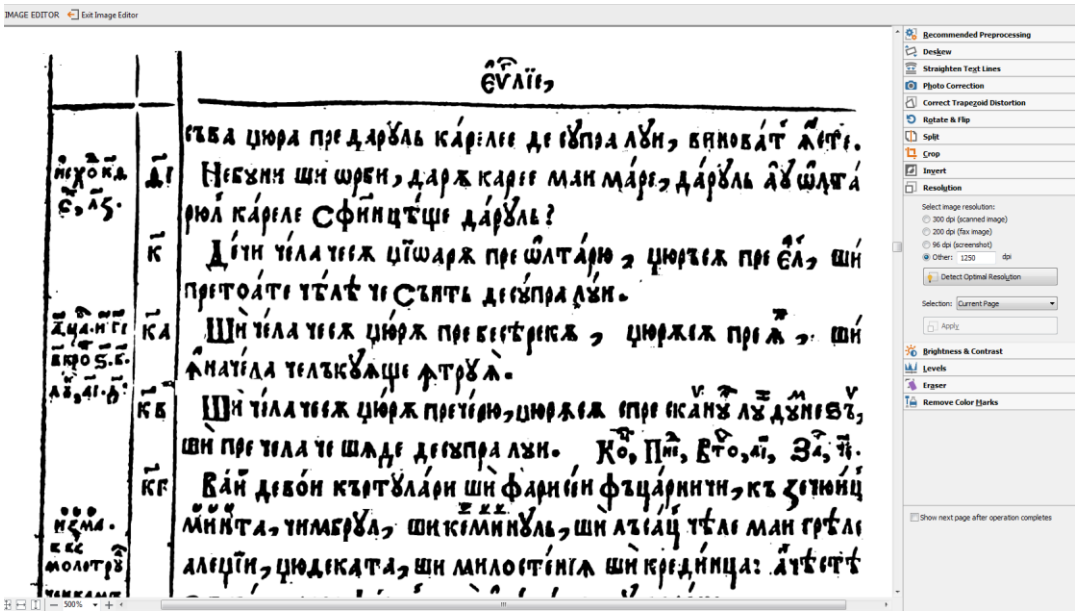



**Figura 2.3 Editorul de imagini integrat în FR 12.**

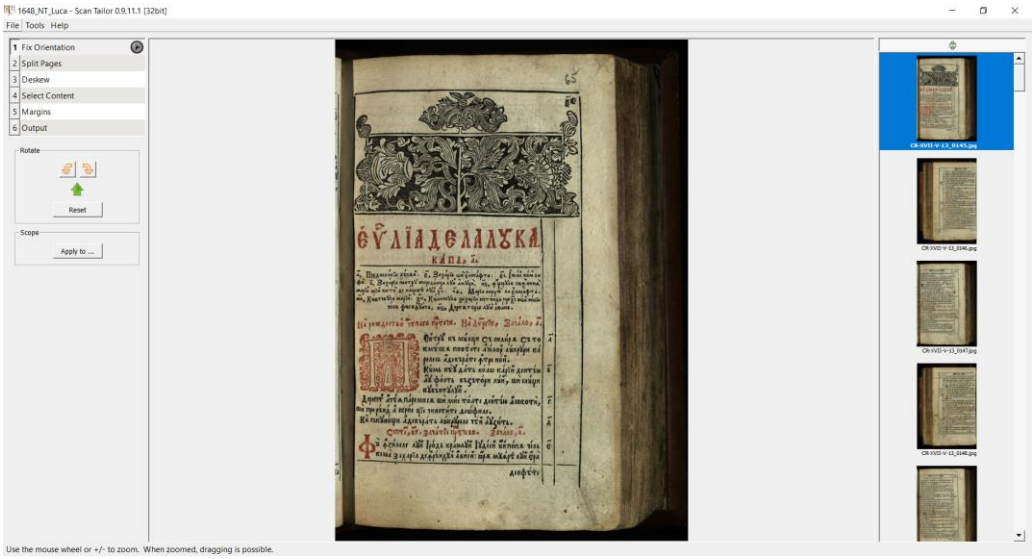



**Figura 2.4. Document din 1648 cu Evanghelia după Luca din Noul Testament încărcat în Scan Tailor pentru preprocesarea imaginii.**

Având un document constituit din mai multe pagini, este necesar să le procesăm separat pe fiecare din ele. De cele mai multe ori noi utilizăm acest instrument pentru a curăța imaginea de artefacte nedorite (*pete, neclarități cu regiuni șterse, zgomot descris printr-o mulțime de puncte negre de pe lângă caractere etc.*), și pentru a îndrepta rândurile. Dacă mai multe pagini se află într-o singură imagine, atunci vom aplica opțiunea de separare a paginilor. Tipul de separare este determinat automat, dar poate fi setat manual și poate fi aplicat tuturor paginilor simultan sau unor pagini individuale. Linia de despărțire poate fi, de asemenea, deplasată automat sau manual, dar nu poate fi aplicată altor pagini. Este util să verificăm panoul de previzualizare (partea din dreapta în Figura 2.4) al fiecărei pagini pentru a ne asigura că separarea pe pagini a fost aplicată corect. Spre deosebire de alți pași, „Ieșirea" devine disponibilă numai după ce toate paginile trec prin pasul „Selectarea conținutului". Acest lucru se datorează faptului că dimensiunea paginilor din ieșire depinde una de cealaltă. Prin urmare, este important să cunoaștem dimensiunea finală a paginilor și asta se poate face numai în pasul de selectare a conținutului. La *ieșire* (output) valoarea implicită a rezoluției imaginii este 600 dpi. În aplicațiile noastre vom păstra această rezoluție. În același timp rezoluția documentului poate fi setată în FR 12 înainte de recunoașterea caracterelor.

Un modul important pe care îl utilizăm prin intermediul Scan Tailor este binarizarea. Aici, binarizarea este implementată prin egalizarea iluminării bazată pe lucrarea [117], netezirea Savitzky-Golay[30], binarizarea propriu-zisă bazată pe Metoda lui Otsu și, la final, eliminarea marginilor întrerupte. Eliminarea marginilor întrerupte se referă la utilizarea unei imagini-șablon

[30] https://en.wikipedia.org/wiki/Savitzky%E2%80%93Golay_filter

ca referință, pentru a localiza și îndepărta marginile întrerupte din imaginea de intrare. În acest caz, imaginea-șablon ar reprezenta o margine liniară fără întreruperi, iar algoritmul ar căuta-o pe aceasta în imaginea de intrare și ar înlocui orice margine întreruptă găsită cu o margine similară celei din șablon. Experiența noastră indică, că este oportun să salvăm documentele în format „alb-negru”, deoarece acesta a demonstrat o acuratețe mai bună la OCR. Totuși, aplicarea acestei opțiuni cere o atenție deosebită, deoarece decolorarea poate duce la pierderea unor elemente de text. Acest lucru poate fi compensat într-o anumită măsură prin îngroșarea caracterelor (putem selecta din opțiunile de salvare la ieșire), dar este important să experimentăm pe câteva pagini înainte de a aplica procedura asupra tuturor paginilor.

Preprocesarea cu *Scan Tailor* este un proces supervizat, deoarece la selectarea fiecărei opțiuni de preprocesare un specialist trebuie să seteze parametrii și să pornească fiecare proces în parte. Astfel, imaginile preprocesate sunt salvate într-o mapă pregătită pentru procesarea ulterioară cu FR 12. În următoarea secțiune vom descrie pașii OCR luându-se în considerare etapa de preprocesare a imaginii.

### 2.3. Crearea limbii utilizatorului și adăugarea dicționarului

Adăugarea alfabetului și crearea limbii utilizatorului (*eng.* user language) este un proces de extindere a capacităților softului ABBY FineReader, prevăzut de autorii acestuia și descris în *ghidul de utilizare*[31] FR 12. Totodată, în cazul alfabetelor vechi, precum cel chirilic românesc, apar unele dificultăți, depășirea cărora necesită acțiuni diferite de cele ordinare. Alfabetul chirilic românesc a înregistrat o proprie evoluție. În cele ce urmează, ne vom referi la cel cu 47 de caractere, utilizat în evul mediu. Vom menționa, că din acestea caracterele Ѧ și Ѕ nu există în sistemul de adăugare a alfabetului în FR12 și nici nu pot fi afișate de fonturile din sistemul acestuia. Prin urmare, a fost necesar să le identificăm și să le adaptăm din softuri specializate, precum *BabelMap*[32]. Restul caracterelor pot fi selectate prin fereastra cu alfabetul FR 12 prezentată în Figura 2.5. Tot din această fereastră poate fi adăugat și dicționarul utilizatorului. Dicționarul de cuvinte poate fi lărgit folosind trei metode. Prima metodă presupune transliterarea vocabularelor existente, din alfabetul modern (latin) în cel chirilic, și adăugarea acestora în dicționarul din FR12. Această metodă rezolvă parțial problema, pentru că multe cuvinte din secolele XVII-XVIII nu se

[31] https://help.abbyy.com/assets/en-us/finereader/12/Users_Guide.pdf
[32] https://www.babelstone.co.uk/Unicode/babelmap.html

mai întâlnesc în vocabularul modern al limbii române și, vice-versa, multe cuvinte din vocabularul modern ar fi de prisos într-un sistem care recunoaște documente tipărite în sec. XVII.

Cea de-a doua metodă presupune crearea dicționarului din textul documentului care a fost deja recunoscut. Prin urmare, toate cuvintele din textul obținut după OCR, se corectează manual și se adaugă la „*dicționarul utilizatorului*” prezentat în Figura 2.6.

Cea de-a treia metodă de adăugare a cuvintelor în dicționarul utilizatorului se bazează, de asemenea, pe faptul că unele porțiuni ale documentului au fost recunoscute. Din fereastra cu textul recunoscut (Figura 2.7), când are loc procesul de verificare ortografică, există posibilitatea de includere a cuvântului subliniat cu linie roșie, în dicționar.

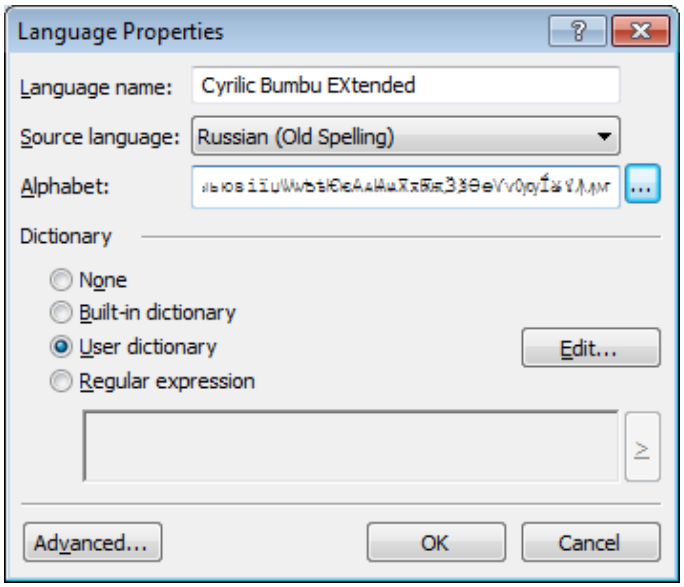



**Figura 2.5. Fereastra de adăugare a alfabetului și creare a limbii utilizatorului în FR 12.**

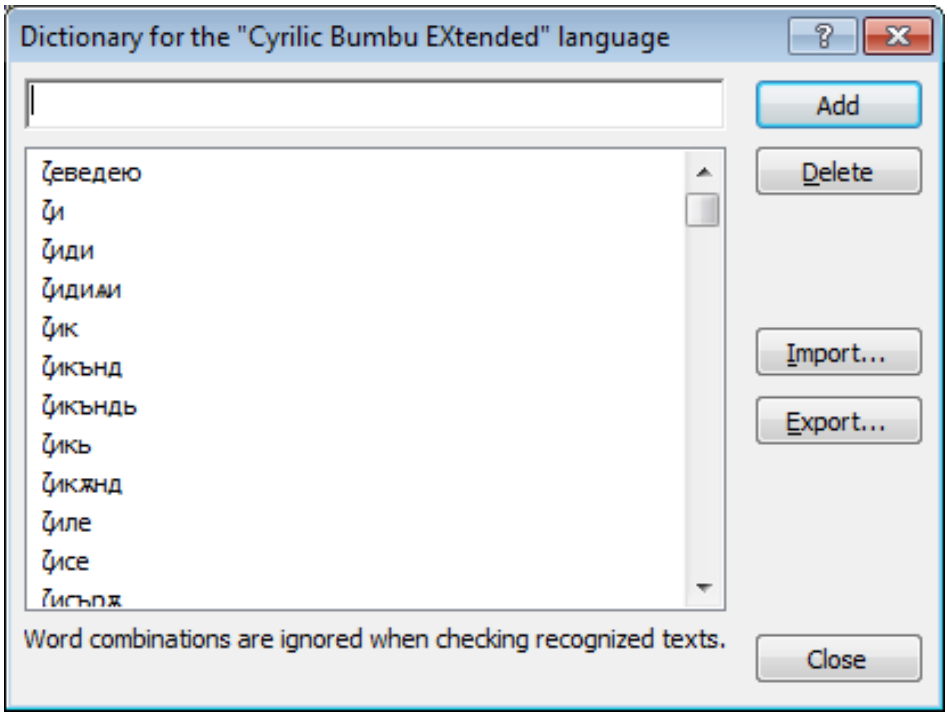



**Figura 2.6. Fereastra de adăugare a dicționarului de cuvinte în FR 12.**

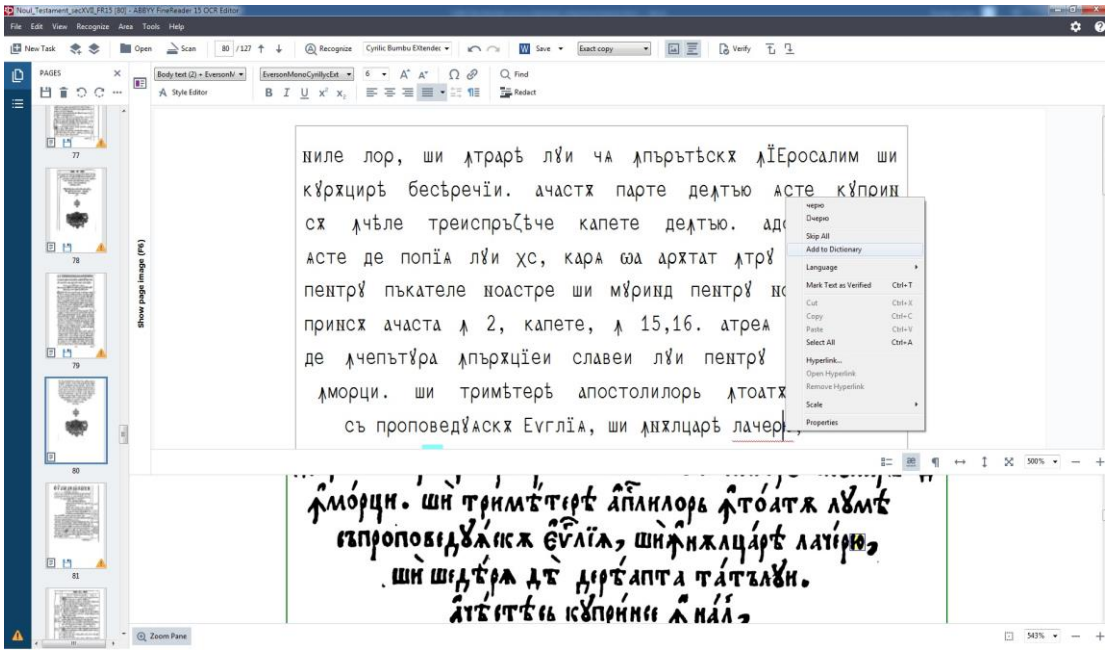


**Figura 2.7. Fereastra FR 15 cu textul recunoscut pe centru. Aici, dacă facem click dreapta pe cuvântul subliniat cu roșu, printre opțiuni apare „*Add to Dictionary*” ceea ce înseamnă că putem adăuga cuvântul subliniat în dicționarul utilizatorului.**

## 2.4. Descrierea procesului de instruire cu FR12, crearea șabloanelor

Un șablon este o mulțime de perechi formată din caracterul tipărit și caracterul digital, care este utilizat de calculator. Șabloanele pot fi create pe parcursul instruirii.

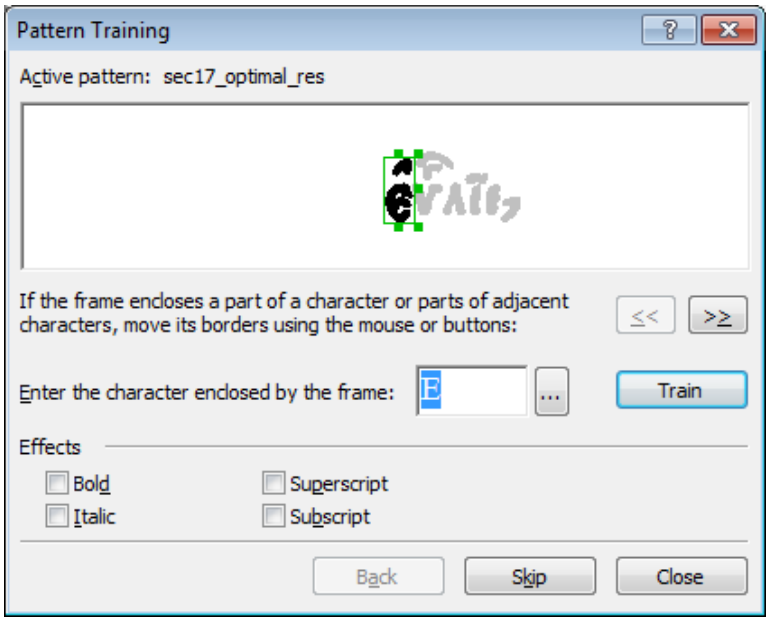



**Figura 2.8. Fereastra FR de învățare a șabloanelor.**

Procesul de instruire a FR12 cu documente românești tipărite în secolul XVII se poate realiza în modul următor: se recunosc două sau mai multe pagini ale documentului prin procedura de instruire supervizată prezentată în Figura 2.8, și, ulterior se creează un șablon. Acest șablon este folosit în continuare drept sursă de informație adițională, pentru a ajuta la recunoașterea textului rămas.

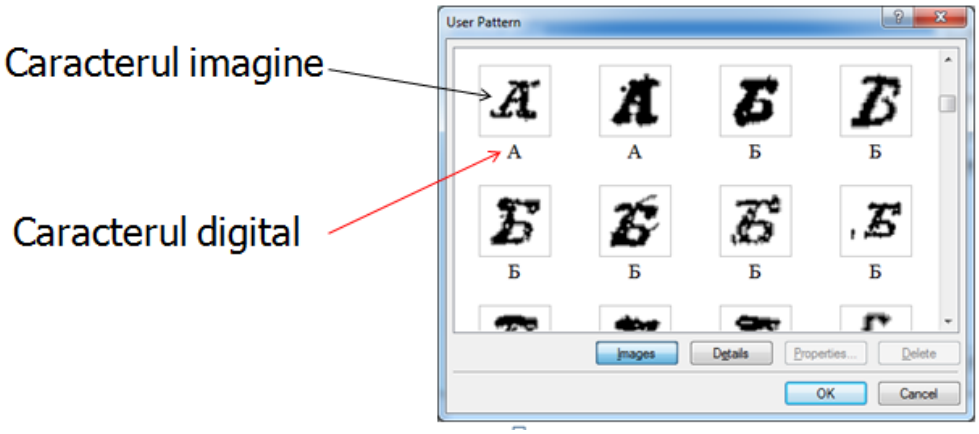



**Figura 2.9. Fereastră din FR 12 cu perechi formate din „*caracterul imagine*” (caracterul din documentul original preprocesat) și „*caracterul digital*” (caracterul UNICODE). Aceste perechi formează un șablon - șablonul utilizatorului.**

Uneori, două sau mai multe caractere pot fi unite între ele, recunoașterea fiecărui caracter în parte devenind practic imposibilă. În acest caz ele vor fi recunoscute împreună într-o singură căsuță de delimitare, în calitate de un caracter compus, numit ligatură. În documentele românești din secolul XVII avem parte de un tip mai deosebit de ligaturi, și anume „*ligaturi verticale*" – set de caractere care nu se unesc explicit și stau una deasupra alteia (vezi Figura 2.10). Exemple de astfel de combinații se întâlnesc foarte des în documentele studiate.

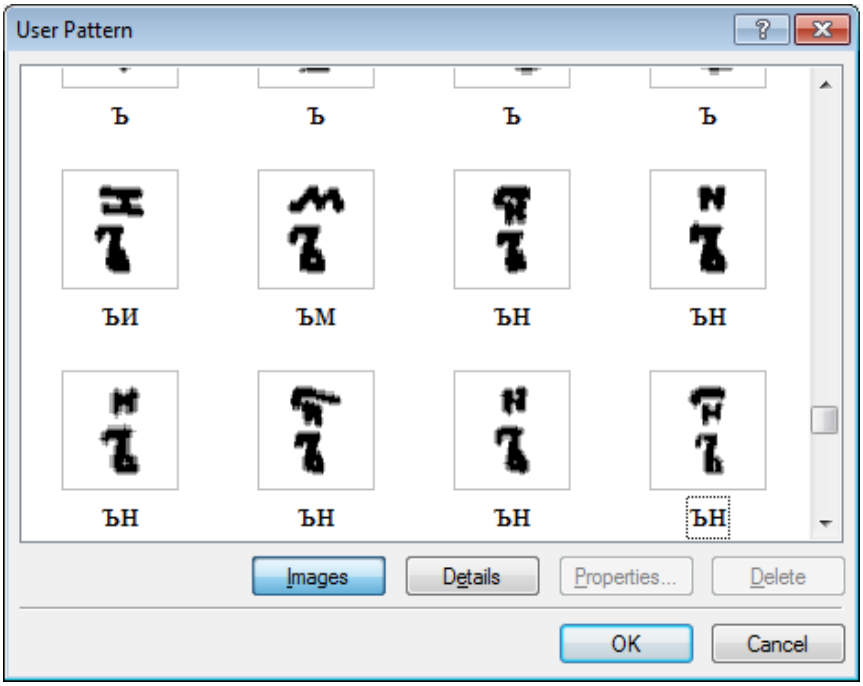



**Figura 2.10. Seturi de ligaturi.**

## 2.5. Modele OCR aplicate pe texte tipărite în secolul XVII

Prin tipăriturile din secolul XVII s-a amplificat și acțiunea de introducere a limbii române în Biserică, adică înlocuirea limbii slavone ca limbă liturgică cu limba română, o realizare deosebit de importantă, finalizată ireversibil la începutul sec. XVIII. În pofida faptului că limba română face parte din grupul celor bazate pe limba latină, grafia în care se tipăreau textele era cea chirilică (slavonă). Luând în considerare că majoritatea tiparelor se aflau în locașuri sfinte (biserici, mănăstiri etc), majoritatea documentelor și cărților tipărite, de asemenea, aveau teme religioase.

Sursa principală de documente românești digitizate din secolul XVII este Biblioteca Digitală a României. Arhiva acestei biblioteci include circa 490 de cărți vechi românești scanate, dintre care 80 de cărți sunt din secolul XVII.

Studiul de caz al recunoașterii optice a cărților din secolul XVII, descris în teză, a fost făcut pe cartea "Noul Testament" tipărită în anul 1648, la Bălgrad (Alba-Iulia) cu tipar negru și roșu, cu

34 de rânduri încadrate pe pagină. Prima copertă este ornamentată cu un chenar cu motive vegetale, realizat prin presare. În centru este reprezentată scena Răstignirii.

Noul Testament conține 682 de pagini, iar primele aprox. 270 de pagini alcătuiesc cele 4 evanghelii:

- Evanghelia după Matei;
- Evanghelia după Marcu;
- Evanghelia după Luca;
- Evanghelia după Ioan;

Recunoașterea optică a caracterelor fost aplicată pe evangheliile după Matei, Marcu, Luca și Ioan care, împreună, constituie 267 de pagini. După preprocesarea cu Scan Tailor, a fost rulată preprocesarea implicită din FR 12. Un aspect comun al cărților tipărite în secolul XVII este scrierea anumitor litere deasupra altei litere exemplificate în Figura 2.11. Acest lucru se poate datora faptului că în procesul de tipărire se omiteau unele litere, iar apoi acestea erau scrise deasupra literei precedente. Cele mai frecvente consoane omise sunt: ***с, д, ш, н, м, л, к, р, х, ц, ч,*** iar vocale: ***а, и, ь***.

**Figura 2.11. Reprezentarea ligaturilor verticale din document (evidențiate prin încercuire).**

De asemenea, sunt folosite abrevieri care folosesc tilde sau alte semne diacritice (Figura 2.12), la fel cum sunt scrise și cifrele (Figura 2.13). Abrevieri se foloseau îndeosebi pentru numele proprii, precum Isus (ЇС), Dumnezeu (Дмнеζъѵ). Luându-se în considerare acest aspect, noi vom mări rezoluția considerabil (peste 1200 DPI) astfel încât să avem tot cadrul unui caracter în procesul de instruire.

**Figura 2.12. Reprezentarea abrevierii *ЇС* (Isus) din document.**

**Figura 2.13. Reprezentarea cifrei 24.**

În procesul de instruire a șabloanelor, cifrele reprezentate prin litere cu tilde deasupra acestora în textul original, au fost înlocuite cu cifre arabe (vezi Figura 2.14).

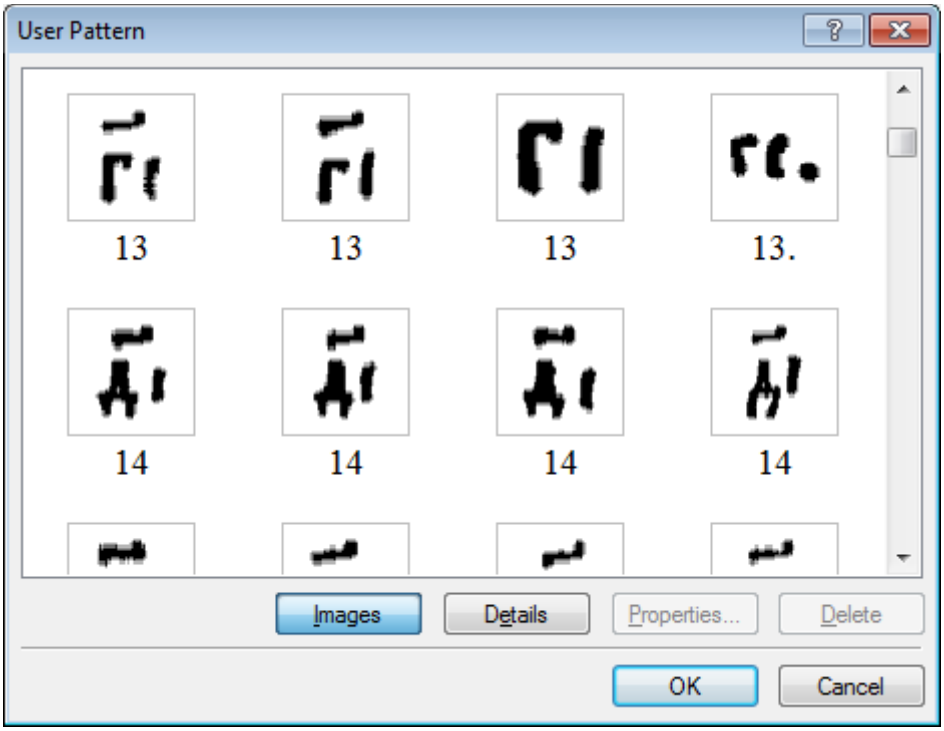



**Figura 2.14. Șablon pentru recunoașterea cifrelor din Noul Testament tipărit în 1648 la Bălgrad.**

O altă caracteristică a cărților și documentelor din secolul XVII este faptul că multe cuvinte sunt tipărite împreună. Din aceste cuvinte compuse, majoritatea sunt combinații din prepoziții/articole unite cu alte părți de vorbire. În Figura 2.15 este arătat un exemplu de cuvânt compus (***алдоиле*** echivalent în prezent, cu expresia ***al doilea***). Acest lucru complică etapa de extindere a dicționarului care servește drept suport la recunoașterea optică. În acest sens, ar fi bine ca toate aceste cuvinte să fie despărțite. Pentru a automatiza acest proces a fost creat un mic program care desparte aceste cuvinte. La baza acestui program stă un vocabular de cuvinte creat manual din documentele deja recunoscute.

În momentul de față dicționarul de cuvinte utilizat la recunoașterea optică a caracterelor pentru textele din secolul XVII conține mai mult de 7500 de cuvinte.

**Figura 2.15.** Exemple de cuvinte scrise împreună (încercuite).

Ținând cont de faptul că în secolul XVII pentru tipar erau utilizate 47 de caractere ale alfabetului chirilic românesc, la etapa de instruire avem nevoie de o tastatură care va conține toate aceste caractere. Ca urmare a fost elaborată o tastatură virtuală (vezi Figura 2.16) pentru sistemele de operare Windows 7, 8, 10 utilizând instrumentul *Microsoft Keyboard Layout Creator*[33], integrând toate aceste caractere.


**Figura 2.16. Tastatura virtuală cu alfabetul chirilic românesc.**

În acest subcapitol au fost descrise unele proprietăți caracteristice recunoașterii documentelor din secolul XVII. În următorul subcapitol vom descrie procesul și rezultatele evaluării OCR pentru documente chirilice românești din secolul XVII.

## 2.6. Evaluarea OCR a documentelor din secolul XVII

La evaluarea OCR a documentelor din secolul XVII tipărite cu caractere chirilice românești vom folosi ABBYY FineReader PDF 15 OCR Editor (în continuare FR 15).

În acest scop a fost pregătit setul de date cu pagini din Noul Testament (1648). Pentru setul de antrenare au fost selectate 10 pagini. Pentru setul de testare au fost selectate 5 pagini care includ aproximativ 7090 caractere și circa 1280 cuvinte. O singură pagină din setul de pagini de testare conține în medie 1400 de caractere și 260 de cuvinte. Aceste pagini deja au fost recunoscute, verificate și corectate manual într-un proces anterior de antrenare, care nu a fost supus unei evaluări cu măsurarea strictă a acurateței. De asemenea, la unele experimente am folosit un dicționar format din 4582 de cuvinte extrase anterior din pagini recunoscute din secolele XVII-XVIII. Cuvintele din dicționar au fost verificate și corectate înainte de a fi adăugate în FR 15.

La această evaluare criteriile luate în considerare au fost acuratețea OCR atât la nivel de caracter, cât și la nivel de cuvânt. Rezultatele evaluării la un anumit număr de pagini de testare/antrenare vor fi rezumate pentru a obține o acuratețe OCR generală. Acuratețea generală (*AG*) pentru un anumit experiment va fi calculată după formula descrisă în lucrarea [118]:

[33] https://www.microsoft.com/en-us/download/details.aspx?id=102134

$$AG = \frac{\sum_1^n c_i}{\sum_1^n t_i} \quad (2.1),$$

unde *n* este numărul de pagini de testare din fiecare experiment în parte, $c_i$ - numărul de caractere sau cuvinte recunoscute corect dintr-o pagina arbitrară, iar $t_i$ este numărul total de caractere/cuvinte din setul de testare folosit în experimente.

Rezultatele evaluării OCR sunt prezentate în Tabelul 2.1. Fiecare rând din tabel reprezintă un experiment efectuat pe un anumit set de date de instruire și testare formate dintr-un număr definit de pagini. Tabelul este compus din următoarele coloane:

- Experiment - care include numărul de pagini folosite la instruire și numărul de pagini folosite la testare;
- Acuratețea OCR la nivel de caracter cu utilizarea dicționarului de cuvinte;
- Acuratețea OCR la nivel de cuvânt cu utilizarea dicționarului de cuvinte;
- Acuratețea OCR la nivel de caracter fără dicționar;
- Acuratețea OCR la nivel de cuvânt fără dicționar.

Vom menționa faptul că evaluarea acurateței este efectuată prin doua metode. Prima metoda este manuală și implică numărarea caracterelor/cuvintelor greșite dintr-o singură pagină din setul de testare de către persoană. Cea de-a doua metodă compară automat diferențele dintre documentul verificat apriori și documentul recunoscut de către modelul nou antrenat.

În continuare vom antrena iterativ un model cu 1, 2, 5, 7 pagini. De asemenea, la fiecare etapă vom testa modelul antrenat împreună cu dicționarul de cuvinte, dar vom face experimente și fără implicarea dicționarului în procesul de recunoaștere.

Primul experiment. Am antrenat modelul FR 15 cu o pagină din setul de antrenare. Din această pagină au fost extrase 662 de glife în procesul de antrenare utilizând interfața GUI oferită de FR 15. Cele mai frecvente caractere sunt: '**и**' și '**а**' care s-au întâlnit de 51 și 48 de ori respectiv, iar numărul minim de glife îl reprezintă caracterul '**ѳ**' care s-a întâlnit doar o singura data în setul de antrenare. La aceasta iterație am evaluat modelul OCR, cu și fără dicționar de cuvinte. Cu dicționar, am numărat 132 de caractere greșite din totalul de 1460 de caractere, iar din 256 de cuvinte în total, 69 au cel puțin un caracter greșit. Fără dicționarul de cuvinte, am constatat prezența a 171 de caractere greșite și a 108 cuvinte care au cel puțin un caracter greșit.

Al doilea experiment. Am antrenat modelul FR 15 cu 2 pagini din setul de antrenare. Din aceste pagini au fost extrase 1275 de glife. Glifa '**а**' apare de 95 de ori în acest set de antrenare, iar în medie sunt peste 30 de glife per caracter. La recunoașterea cu dicționar a unei pagini din setul de testare, am numărat 99 de caractere greșite din 1460 de caractere, iar 56 de cuvinte au cel puțin

un caracter greșit. Fără dicționarul de cuvinte, au fost evidențiate 134 de caractere greșite și, respectiv, 76 de cuvinte greșite din 256 în total.

Al treilea experiment. Am antrenat, în continuare, modelul OCR cu 5 pagini din setul de antrenare conținând peste 2500 de glife. La recunoașterea cu dicționar, din 1460 de caractere testate 73 au fost recunoscute greșit, iar din 256 de cuvinte evaluate, 50 s-au dovedit a fi greșite. Fără dicționar, s-a constatat prezența a 86 de caractere identificate greșit, precum și a 69 de cuvinte cu erori. Majoritatea erorilor înregistrate în acest experiment sunt cauzate de ligaturi, mai exact, 44 din 56 de ligaturi fiind eronate.

Al patrulea experiment. Modelul OCR a fost antrenat cu 7 pagini din setul de antrenare. Numărul de glife din set este peste 3600. Cu dicționar, 59 de caractere din 1460 au fost recunoscute greșit, iar 52 de cuvinte din 256 au cel puțin un caracter greșit. În comparație cu experimentul numărul 3, acuratețea la nivel de caractere dobândita în acest experiment a continuat să crească cu aproximativ 1%, însă acuratețea la nivel de cuvinte a rămas la același nivel, ba chiar avem cu 2 cuvinte greșite mai mult ca în experimentul anterior, în cazul utilizării dicționarului.

**Tabelul 2.1. Acuratețea OCR la recunoașterea documentelor din secolul XVII.**

| **Experimente** | | | **Cu dicționar** | | **Fără dicționar** | |
|---|---|---|---|---|---|---|
| **Nr.** | **Antrenare** | **Testare** | $A_{ch}$ | $A_{cuv}$ | $A_{ch}$ | $A_{cuv}$ |
| 1 | 1 pagina (662 glife); | 1 pagină; | 0.91 | 0.73 | 0.88 | 0.57 |
| 2 | 2 pagini (1275 glife); | 1 pagină; | 0.93 | 0.78 | 0.90 | 0.70 |
| 3 | 5 pagini (2540 glife); | 1 pagină; | 0.95 | 0.80 | 0.94 | 0.73 |
| 4 | 7 pagini (3668 glife); | 1 pagină; | 0.96 | 0.796 | 0.95 | 0.75 |

Modul de antrenare și evaluare a modelului OCR utilizând FR 15 la recunoașterea caracterelor chirilice românești din secolul XVII, prezentat mai sus, ne permite să concludem, că respectivul sistem software conține componentele majore necesare pentru instruire, fiind posibil de a-l aplica și pentru cazul tipăriturilor vechi românești (în cazul nostru – din sec. XVII), care nu erau prevăzute în setul inițial. O abordare bazată pe astfel de instrumente dotate cu o interfață grafică cu utilizatorul permite, de asemenea, și utilizatorilor neavansați să instruiască motoare OCR și să participe în proiecte de digitizare.

Abordarea instruirii a fost evaluată utilizând mai multe pagini de antrenare și testare într-un proces iterativ, de la o pagină de antrenare până la 7 pagini. Pe parcursul creșterii numărului datelor de antrenare, s-au observat îmbunătățiri semnificative ale acurateței modelului. Exemple

de recunoaștere, transliterare și aliniere ale unui document din secolul XVII sunt prezentate în Figurile 2.17-2.20.

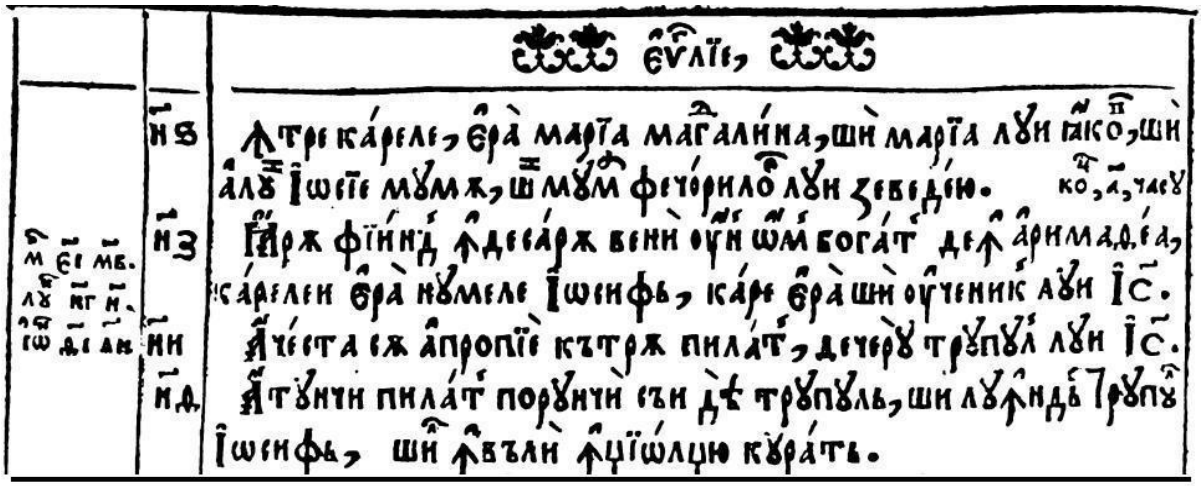


**Figura 2.17. Fragment din Noul Testament tipărit în anul 1648 la Bălgrad.**

Єvгліє,

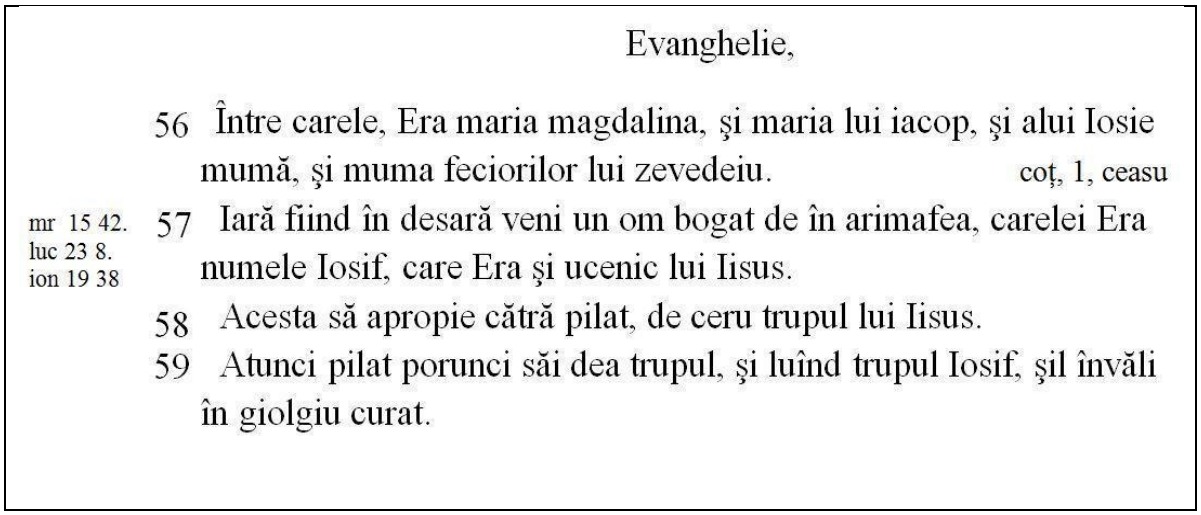
56 Лтре кареле, єра маріа магдалина, ши маріа лзи іакоп, ши алзи Іωсіе
мзмж, ши мзма фечорилор лзи Зеведею. коц, 1, чиєз

мр 15 42. 57 Шдеж фіінд д десдеж вени оун шм богат де д аримадеа, карелен єра
лзк 23 8. нзмеле Іωсифь, каре єра ши оученик лзи Іс.
їωн 19 38

58 Дчеста сж апропіе кътеж пилат, де черз трзпзл лзи Іс.

59 Дтзнчи пилат порзнчи сзи дѣ трзпзль, ши лзндь трзпзл Іωсифь, ши
двзли д цїшлцю кзрать.

**Figura 2.18. Documentul din figura 2.17 după etapa de OCR.**


Evanghelie,

56 Între carele, Era maria magdalina, şi maria lui iacop, şi alui Iosie
mumă, şi muma feciorilor lui zevedeiu. coţ, 1, ceasu

mr 15 42. luc 23 8. ion 19 38 57 Iară fiind în desară veni un om bogat de în arimafea, carelei Era
numele Iosif, care Era şi ucenic lui Iisus.

58 Acesta să apropie cătră pilat, de ceru trupul lui Iisus.

59 Atunci pilat porunci săi dea trupul, şi luînd trupul Iosif, şil învăli
în giolgiu curat.

**Figura 2.19. Textul din Figura 2.6.2 după etapa de transliterare.**

56. Intre care erau Maria Magdalena si Maria, mama lui Iacob si a lui Iosif, si mama fiilor lui Zevedeu.

57. Iar in amurgul zilei a venit un om bogat din Arimateea, cu numele Iosif, care era si el ucenic al lui Iisus.

58. Acesta, ducandu-se la Pilat, a cerut trupul lui Iisus. Atunci Pilat a poruncit sa i se dea.

59. Si Iosif, luand trupul, l-a infasurat in giulgiu curat.

**Figura 2.20. Fragment din *Evanghelia după Matei*, varianta actuală aliniată la textul din Figura 2.19.**

## 2.7. Clasificarea fonturilor din secolul XVII

Tiparniţele din secolul XVII foloseau mai multe fonturi sau mai bine zis, mai multe seturi de caractere distincte la tipărirea documentelor, dar dintre acestea, deocamdată, se disting două fonturi total diferite, atât după stilul scrierii/tipăriturii cât și după utilizarea caracterelor [29, 30].

Problema identificării fontului într-un document tipărit în secolul XVII poate fi formulată în felul următor: *Se dă un document X tipărit în secolul XVII cu caractere chirilice româneşti şi un set N de modele OCR antrenate pe documente din această perioadă. Să se aleagă cel mai potrivit model M din setul N pentru recunoaşterea documentului X* [30].

O soluție ar fi să recunoaștem o mostră (o pagină/un fragment de pagină) din documentul *X* cu toate șabloanele antrenate în FR 12 pe documente din sec. XVII și în baza rezultatelor obținute să alegem șablonul care oferă acuratețea cea mai mare. Această soluție este ușor de implementat,

dar complexitatea de timp este prea mare, întrucât trebuie să încărcăm pe rând fiecare model *M* în parte. Timpul de încărcare a unui model plus recunoașterea mostrei poate depăși 2 minute în dependență de mărimea paginii. Pentru 5 modele OCR diferite, vom fi nevoiți să așteptăm aproximativ 10 minute ca să găsim cel mai potrivit model *M*.

Dat fiind faptul, că numărul primelor tipografii nu era prea mare, am putea aborda o altă soluție directă, unde toate modelele OCR sunt clasificate după *Tipografii* și în care utilizatorul poate alege tipografia. Această soluție a fost implementată și descrisă în următoarea secțiune.

**Program de selectare a modelului OCR potrivit în funcție de tipografie**

În acest program [29], utilizatorul poate să aleagă secolul, iar din următoarele opțiuni ar putea alege una dintre regiunile unde se folosea grafia chirilică românească, și anume: ***Iași, București, Târgoviște, Bălgrad (Alba Iulia), Uniev (Cernăuți), Sas Sebeș, Snagov, Buzău.*** Vom continua cu selectarea tiparniței din una din aceste regiuni. Așadar pentru ***Iași*** sunt următoarele tiparnițe:

1. *Tipariul cel Domnesc;*
2. *Casa Sfintei Mitropolii;*
3. *Tiparnița Tărâi;*

La ***București*** erau următoarele tipografii:

1. *Scaunul Mitropolii Bucureștilor;*
2. *Tipografia Domnească;*

***Bălgradul*** avea următoarele tiparnițe:

1. *Tipografia Domnească;*
2. *Mitropolia Bălgradului.*

Celelalte regiuni aveau câte o singură tiparniță, respectiv vom prezenta denumirile acestora în original:

- *Târgoviște – Sfânta Mitropolie a Târgoviștii;*
- *Uniev – Sfînta Mănăstire Uniev;*
- *Sas Sebeș – Tipograf[ia] Noao;*
- *Snagov - Tipografiia Domnească în Sfânta Mănăstire în Snagov;*
- *Buzău - Tipografiia Domnească, la Episcupiia dela Buzău;*

Documentele și cărțile care au fost tipărite în aceste tipografii vor putea fi recunoscute folosind cel mai potrivit șablon. Fereastra principală din FR 12 se va deschide cu șablonul setat, iar mai departe va avea loc procesul de recunoaștere optică a caracterelor.

În continuare, în figura 2.21 este prezentată interfața grafică a aplicației de selectare a celui mai potrivit șablon de recunoaștere. Aplicația „Model Selector” este scrisă în întregime în Java și lucrează cu orice versiune FR instalată.

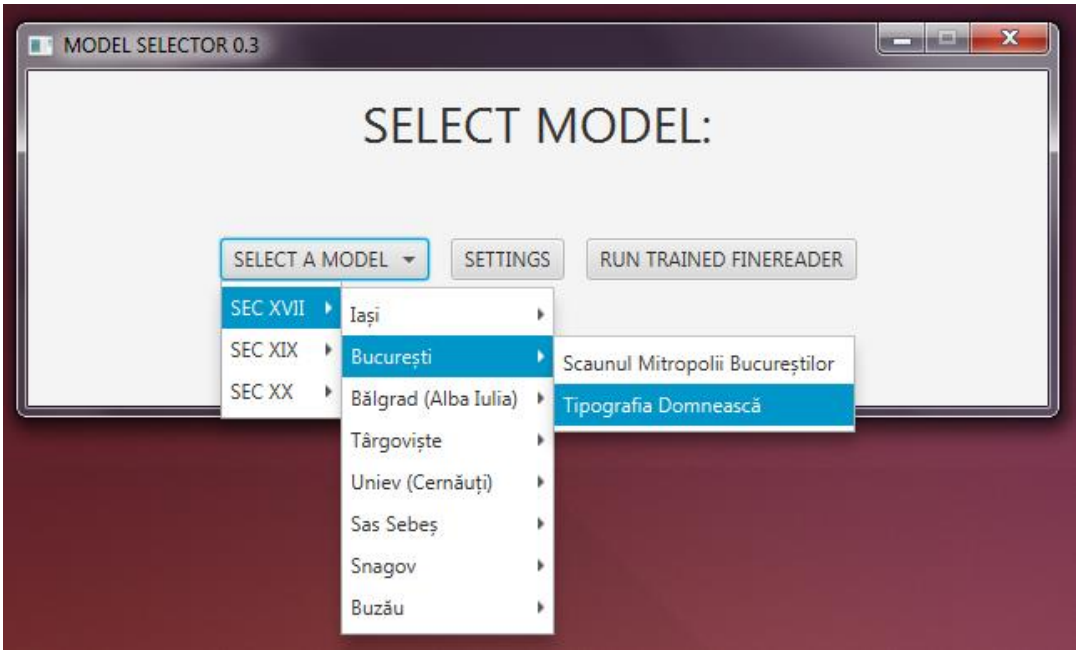



**Figura 2.21.** Fereastra principală a aplicației Model Selector.

Cea de-a treia soluție ar fi să învățăm o rețea neurală să clasifice o mostră din documentul X având la bază mostre din mai multe documente românești tipărite în anii 1640-1700, la diferite tipografii. Vom utiliza o rețea neurală, care va folosi la învățare un set de date format din perechi de forma <caracter din imagine, clasă>. În continuare vom descrie implementarea acestei soluții.


**Figura 2.22.** Texte tipărite în secolul XVII în alfabet chirilic român la două tipografii diferite, fiecare cu fonturi distincte [30].

În Figura 2.22, putem observa două pagini din două cărți din secolul XVII care au fost tipărite cu două fonturi diferite - cu două seturi de caractere diferite. Se poate observa că cele două texte au stilurile diferite și pe lângă aceasta sunt utilizate caractere cu forme foarte distincte. Dacă vom alege un șablon care a fost instruit pe primul text și îl vom aplica pe cel de-al doilea text afișat în figura 2.22, atunci vom obține un rezultat cu o rată mare de erori. În acest caz, cea mai potrivită soluție ar fi crearea unui nou șablon instruit pe cel de-al doilea text. În cele ce urmează vom descrie modul de soluționare a acestei probleme prin crearea unei rețele neurale, antrenate să efectueze clasificare a două fonturi distincte, folosite în secolul XVII la tipărirea documentelor în limba română.

### Clasificarea fonturilor utilizând rețele neurale

În cele ce urmează vom descrie soluționarea problemei de clasificare a caracterelor din secolul XVII. Putem constata din start existența mai multor instrumente gratuite utile, care pot contribui la dezvoltarea modulului respectiv. Și, desigur, unul din aspectele principale specific oricărei aplicații bazate pe rețele neurale, îl constituie crearea setului de date pentru antrenarea algoritmilor de învățare automată.

#### *Crearea setului de date*

Setul de date va fi creat din 10 cărți scanate, selectate din Biblioteca Digitală a României[34], acestea fiind următoarele:

1. „Noulu Testamentu sau Înpacarea” tipărit în 1648 în *Cetatea Bălgradului*.
2. „Liturghie şi rugăciuni” tipărită în 1683 în *Casa Sfintei Mitropolii* de la Iași de mitropolitul Dosoftei.
3. „Parimiile preste an” tiparită în 1683 în *Tiparniţa Ţărâi* de la Iași.
4. „Psaltirea a proroculuı şi înpăratului D[a]v[i]dŭ Cu m[o]l[i]tve la toate Cathizmele Şi cu pashalii de 50 de ani. După orânduiala grecească. Şi la săvârşitŭ exapsalmu” tipărită în 1694 la *Tipografia Domnească în Sfânta Mitropolie* din București.
5. „Carte sau lumină cu dreapte dovediri din dogmele Besearicii Răsăritului, asupra dejghinării Papistaşilorŭ” tipărită în 1699 în *Tipografia Domnească în Sfânta Mănăstire în Snagov*.

[34] http://digitool.bibnat.ro/ (Carte românească veche şi bibliofilă/Sec. XVII)

6. „Pravoslavnica mărturisire a săborniceştii, şi apostoleştii Besearecii Răsăritului Dupre Grecească” tipărită în 1691 *În tipografia Domnească, la Episcupiia de la Buzău* de Petru Movilă, mitropolit al Kievului.
7. „Septembrie-Decembrie Vol. 1” tipărită în 1682 în *Tipografia Sfintei Mitropolii* de la Iași.
8. „Mineiulŭ. Luna lui Dechemvrie” tipărită în 1698 în *Sfânta Episcopie* de la Buzău.
9. „Apostolul cu Dumnezău Svântul Carea întru acesta chip tocmită depre orânduiala grecescului Apostol” tipărită în 1683 în *Scaunul Mitropolii Bucureştilor*.
10. „Sfânta şi Dumnezeiasca Evanghelie” tipărită în 1697 în *Sfânta Mănăstire în Sneagovŭ* din Snagov.

În cele zece cărți selectate s-au observat doua seturi de caractere distincte, folosite la tipărire, pe care le vom numi fonturile *A* si *B*. Astfel, cărțile utilizate pentru formarea setului de date au fost împărțite în două grupuri: cărțile cu nr. 1, 8, 9, 10 grupate în setul cu fontul *A* și cărțile cu nr. 2, 3, 4, 5, 6, 7 grupate în setul *B* (cel de-al doilea font). La formarea setului de date s-a operat în total cu 22 de pagini, inclusiv 13 pagini extrase pentru setul *A* și 9 pagini – pentru setul pentru *B*. În figura 2.23a și 2.23b sunt redate două colaje alcătuite din unele pagini selectate.

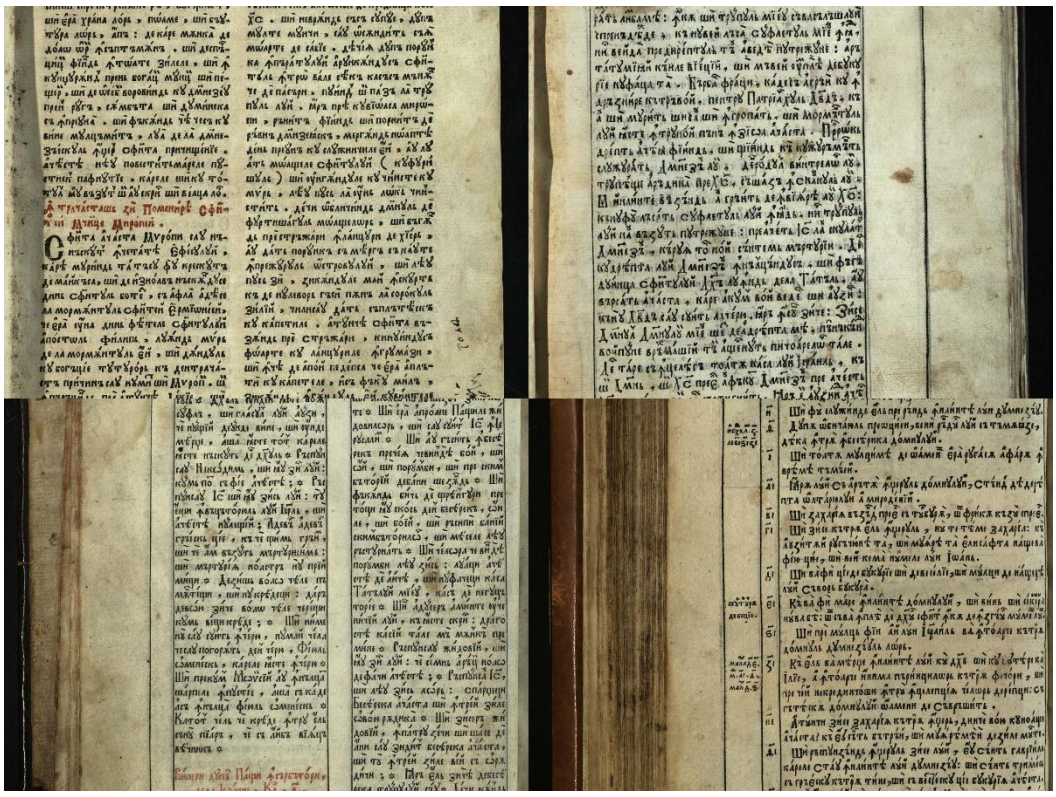


**Figura 2.23a. Colaj din cărțile cu nr. 1, 8, 9, 10 cu setul de caractere *A*.**

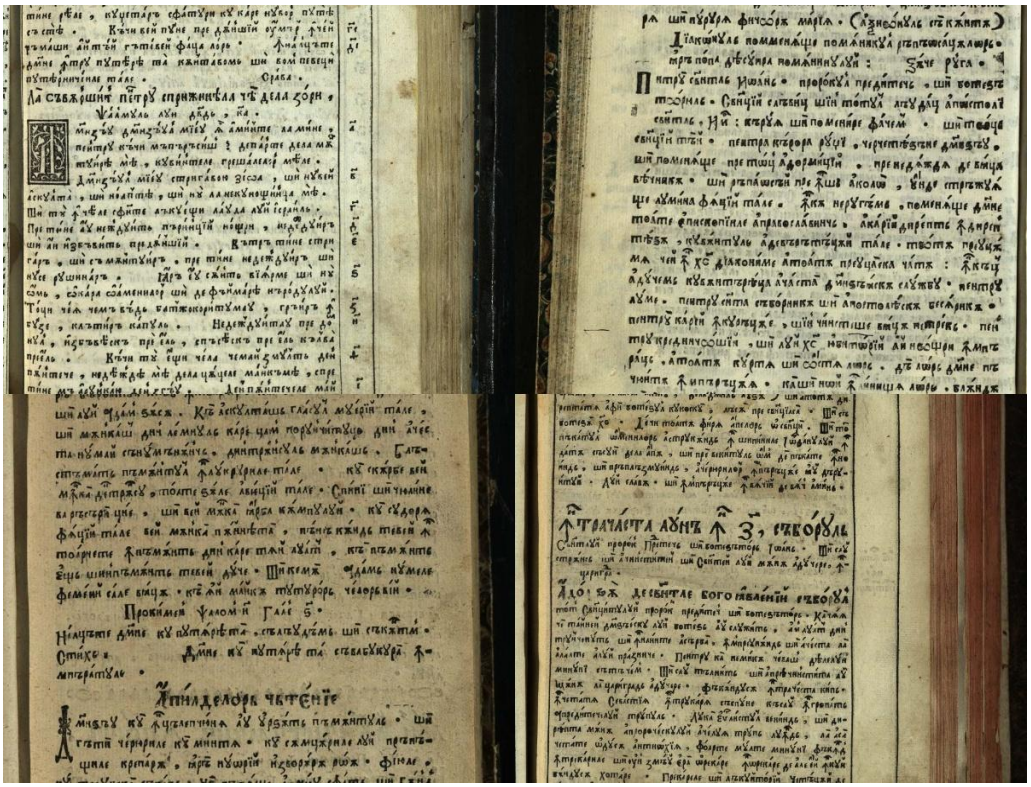


**Figura 2.23b. Colaj din cărțile cu nr. 2, 3, 4, 7 cu setul de caractere *B*.**

Pentru soluționarea problemei de clasificare a fonturilor este necesar să executăm următoarele acțiuni:

1. Segmentarea și decuparea blocurilor cu text din paginile selectate.
2. Detectarea caracterelor individuale din blocurile cu text.
3. Crearea setului de date de antrenare și testare din caracterele detectate în pasul nr. 2 prin clusterizarea/gruparea caracterelor extrase.
4. Antrenarea unei rețele neurale multistrat (RNM) pentru a clasifica caracterele și evaluarea acesteia.
5. Utilizarea algoritmului instruit (modelului) pentru a clasifica caracterele dintr-o imagine nouă.

În cele ce urmează vom descrie în detalii modalitatea de executare a sarcinilor enumerate mai sus.

***Segmentarea și decuparea blocurilor cu text***

Din paginile pregătite pentru setul de date vom segmenta și decupa fragmente de text utilizând *Detectron2*[35] – o platformă pentru detectarea obiectelor, segmentarea și realizarea altor

[35] https://ai.facebook.com/tools/detectron2/

sarcini de recunoaștere vizuală, dezvoltată de Facebook, precum și modelul de segmentare a documentelor pe baza setului de date *PrimaLayout* [36] cu configurarea *mask_rcnn_R_50_FPN_3X* implementate în pachetul *LayouParser*[37].

În modelul cu setul de date *PrimaLayout* porțiunile de text sunt etichetate cu eticheta „TextRegion" (regiune/bloc cu text) care are ID-ul 1. Pe baza acestui ID vom selecta blocurile cu text.

Luând în considerare complexitatea machetei în documentele din secolul XVII și anume: *titlul, versetele din rând nou, cifrele scrise cu litere chirilice, notele de la începutul capitolelor, etc.* vom segmenta și tăia mai mult de un bloc cu text dintr-o singură pagină. Din acest motiv s-ar putea ca două blocuri de text sa conțină aceleași caractere, iar setul nostru de date să conțină drept urmare exemple de antrenare și testare similare. Acest lucru nu prezintă o problemă la etapă inițială, optimizarea setului de date fiind posibil de realizat mai târziu, în pasul 3, la crearea setului de date de antrenare și testare.

La aplicarea instrumentarului de segmentare asupra paginilor selectate vom obține porțiuni cu blocuri de text care ne vor ajuta la decuparea fragmentelor necesare. Un exemplu de segmentare este afișat în Figura 2.24. Din această imagine putem observa că s-au conturat două dreptunghiuri care se intersectează delimitând două blocuri cu text, pe când pagina/imaginea este de fapt constituită dintr-un singur bloc. Dar, după cum am menționat mai sus, la etapa inițială repetarea acelorași caractere provenite din diferite blocuri de text nu este un impediment.

[36] https://www.primaresearch.org/dataset/
[37] https://github.com/Layout-Parser/layout-parser/blob/master/docs/notes/modelzoo.md

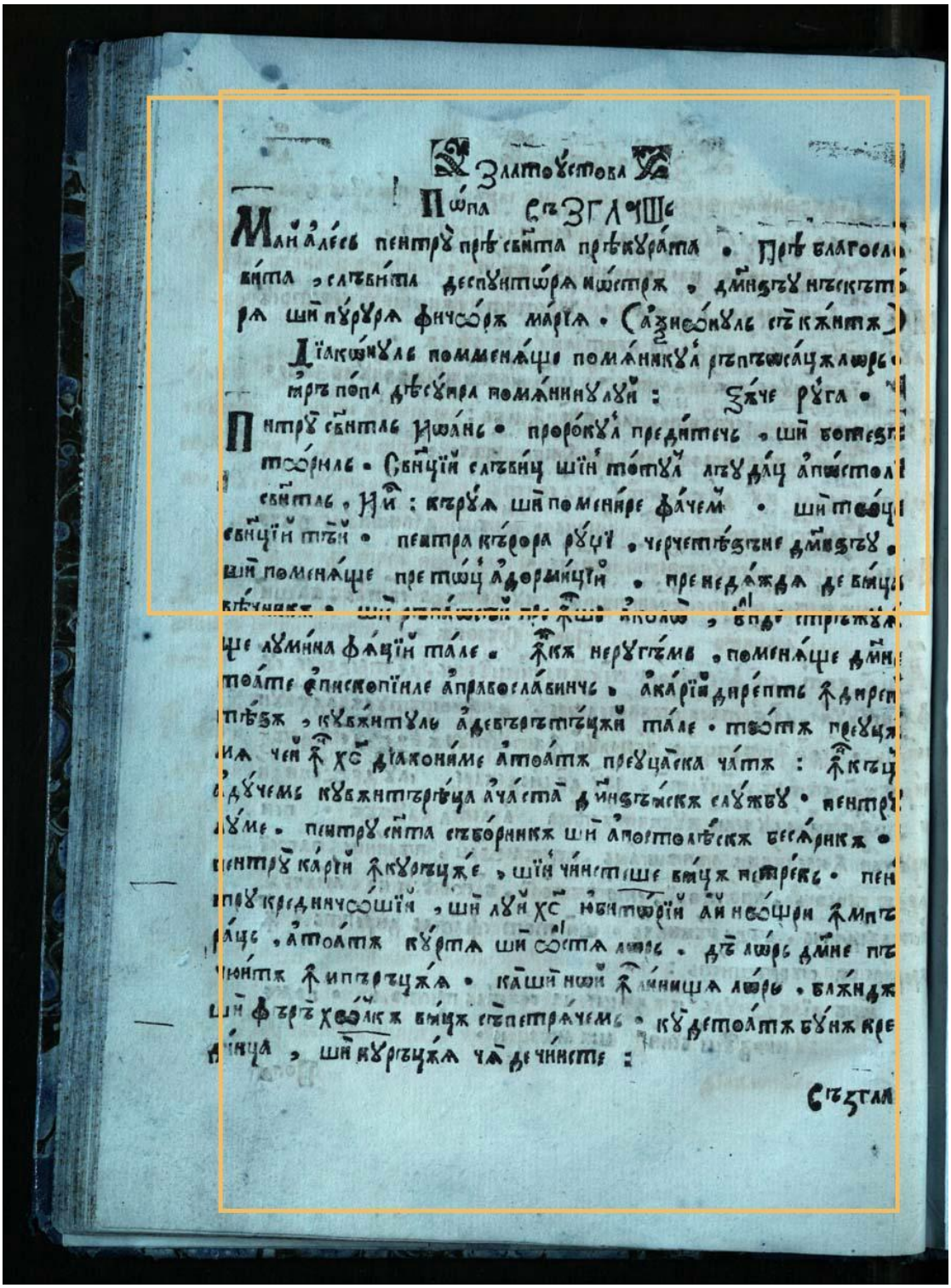


**Figura 2.24. O pagină din cartea nr. 2 care a fost segmentată cu *PrimaLayout.***

După segmentare, blocurile de text urmează a fi decupate și plasate într-o listă de fragmente cu blocuri de text. În medie, s-au obținut câte patru fragmente-imagini cu blocuri de text pentru fiecare din cele 22 de pagini.

La următorul pas din aceste fragmente vom identifica și extrage prin decupare caracterele. Caracterele sunt literele, semnele de punctuație, accente, unele linii din conturul tabelelor, pixeli ale petelor sau deteriorărilor de pagini.

***Detectarea caracterelor individuale din blocurile cu text***

Așadar, următoarea sarcină este de a găsi o modalitate pentru a extrage caractere din fragmentele cu blocuri de text. În pasul următor (pasul 3) putem crea apoi un set de date din literele extrase și, în cele din urmă, să antrenăm un clasificator pe acest set de date.

Intrările în sistemul nostru sunt pagini din documente tipărite în secolul XVII. Aceste intrări pot fi imagini provenite din diferite surse (cameră smartphone sau foto, scaner etc.), având rezoluții diferite. Scopul pe care îl urmărim este ca fiecare imagine, independent de sursa sa, să fie procesată astfel, ca algoritmul folosit la detectarea caracterelor să poată găsi cât mai multe litere. Vom reține, că stocarea imaginilor de către camerele digitale se efectuează în trei canale separate: roșu, verde și albastru (RGB). În cazul nostru aceste trei canale conțin informații redundante, deoarece literele pot fi identificate în fiecare dintre aceste trei canale separat. Prin urmare, vom converti mai întâi toate imaginile în alb-negru, astfel ca în loc de trei canale, să operăm cu unul singur, ceea ce ar trebui să sporească viteza de învățare. Ca urmare, imaginea procesată va consta doar din pixeli alb-negru. Putem apoi să optimizăm în continuare imaginea pentru detectarea literelor. Astfel, adițional la softul existent, a fost elaborată o funcție Python care face conversia imaginilor din RGB în alb-negru cu ajutorul bibliotecii OpenCV.

Pentru detectarea caracterelor vom utiliza metoda *findContours()* din biblioteca openCV. Putem apoi să mapăm casetele (dreptunghiurile) de delimitare ale contururilor găsite de această funcție înapoi pe imaginea RGB originală pentru a vedea ce a fost detectat de fapt. Rezultatul acestei etape de procesare este exemplificat în Figura 2.25.

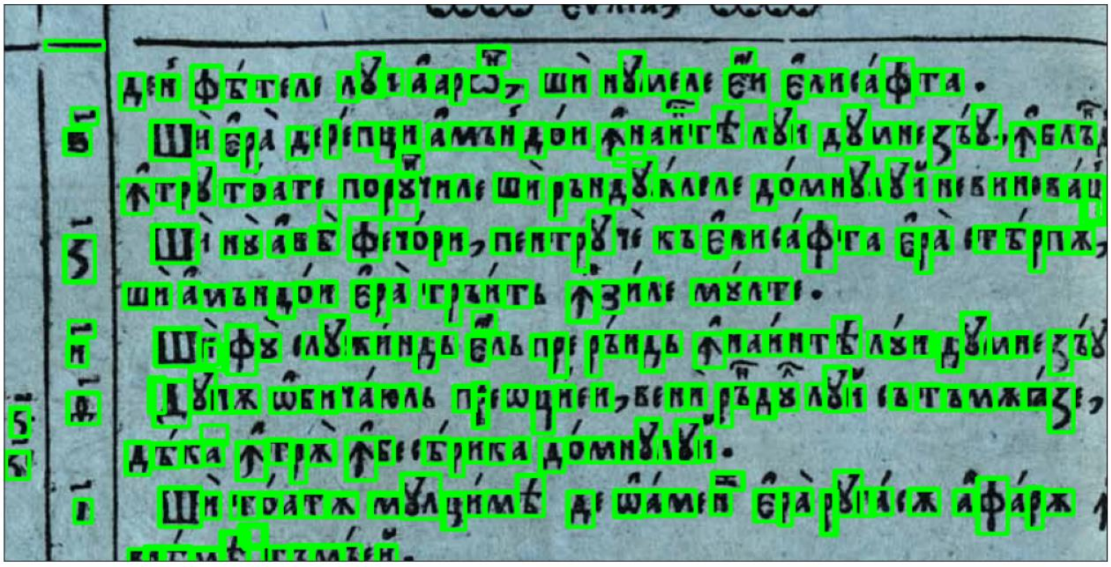


**Figura 2.25. Identificarea contururilor caracterelor într-un fragment cu bloc de text.**

Caracterele identificate au fost decupate și salvate în două dosare, pentru fiecare font în parte. Din 10 pagini din setul *A* am obținut 17155 de caractere, iar din 9 pagini ale setului *B* au fost salvate 8799 de caractere. Diferența dintre numărul de caractere extrase pentru setul *A* și *B* se explică prin dimensiunile caracterelor folosite în cele două fonturi distincte, prin numărul de caractere în pagină, dar și prin numărul de fragmente de blocuri de text decupate. Dintre caracterele extrase, unele ar putea fi elemente de zgomot din imagine - pete, accente fără caracterul de bază (literă), tilde. La pasul următor  vom elimina caracterele inutile și păstra doar literele utilizând modelul de clusterizare *K-Means*[38] și *Principal Component Analysis*[39] (PCA).

***Gruparea caracterelor pentru crearea setului de date de antrenare și testare***

Următoarea sarcină, ce trebuie realizată, constă în eliminarea imaginilor care nu conțin litere și  gruparea prin clusterizare a tuturor imaginilor rămase (aceasta înseamnă că toate literele din colecția „A” intră într-un dosar, toate literele „B” – în alt dosar). Pentru a obține un set de date calitativ vom îmbina procesarea manuală cu cea automatizată prin gruparea datelor cu PCA și K-Means.

Metodele PCA și K-Means se preocupă de sarcini diferite. PCA este utilizat pentru reducerea dimensionalității sau selectarea caracteristicilor atunci când spațiul acestora conține prea multe caracteristici irelevante sau redundante. Scopul este de a găsi dimensiunea intrinsecă a

[38] https://scikit-learn.org/stable/modules/generated/ sklearn.cluster.KMeans.html
[39] https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html

datelor. Pe de altă parte, K-Means este un algoritm de grupare care returnează gruparea naturală a vectorilor caracterelor, pe baza asemănării lor.

Înainte de a începe clusterizarea, trebuie să ne asigurăm că toate imaginile au aceeași dimensiune. După cum putem vedea din figura 2.7.1.4, caracterele au fost salvate cu dimensiuni diferite. Fiecare imagine are dimensiunea casetei sale de delimitare, care variază pe o scară largă. Deci, în primul pas vom redimensiona toate imaginile la dimensiunea de 50x50 pixeli și le vom adăuga pe toate într-o matrice utilizând biblioteca *NumPy*[40]. Apoi, va trebui să reducem dimensiunea caracterelor. Fiecare imagine conține *50x50* (2500) pixeli sau caracteristici. Această cantitate este prea mare pentru un algoritm de clusterizare, vom folosi analiza componentelor principale (PCA) pentru a reduce numărul de dimensiuni de la 2500 la 25.

În urma acestor acțiuni datele sunt pregătite pentru a fi grupate astfel ca setul de date dintr-o stare fără etichete să fie adus într-o stare etichetată. Vom aplica clusterizarea K-Means, aceasta fiind o metodă simplă și rapidă. Singurul lucru pe care trebuie să-l furnizăm este numărul de clustere pe care îl așteptăm. Dacă K-Means ne-ar oferi un rezultat perfect, atunci am obține exact numărul de litere din alfabetul chirilic român - 47, dar este mai puțin probabil să obținem un astfel de rezultat ideal, luând în considerare și unele variații privind diferențele dintre majuscule și minuscule. Fiecare dintre cele 47 de litere poate apărea sub formă de literă mare sau mică, ceea ce înseamnă că în total ne-am putea aștepta și la 47x2 (94) de clustere. De asemenea, vor exista semne de punctuație și zgomote în date, care au fi fost detectate. Prin urmare, este rațional să indicăm parametrul de ieșire egal cu 100 de clustere. Aceasta este mai mult decât avem nevoie, dar va fi mai ușor să îmbinăm clusterele mai târziu decât să le separăm.

Procesul de clusterizare pentru cele circa 25 de mii de caractere a durat mai puțin de un minut, iar rezultatul clusterizării îl vom muta în foldere separate pe baza etichetei clusterelor (Figura 2.26).

[40] https://numpy.org/doc/stable/user/whatisnumpy.html


**Figura 2.26. Foldere cu caracterele grupate în clustere.**

După clusterizarea automată, urmează procesarea manuală, care constă în următoarele acțiuni: mutarea imaginilor clasificate greșit în dosarele corecte și îmbinarea dosarelor cu litere identice, identificarea zgomotelor etc. Vom observa, că unele clustere sunt formate din părți de caractere (*unghiuri, curbe, linii*) (vezi Figura 2.27), iar altele sunt formate din litere ale alfabetului chirilic românesc (vezi Figura 2.28 și Figura 2.29).

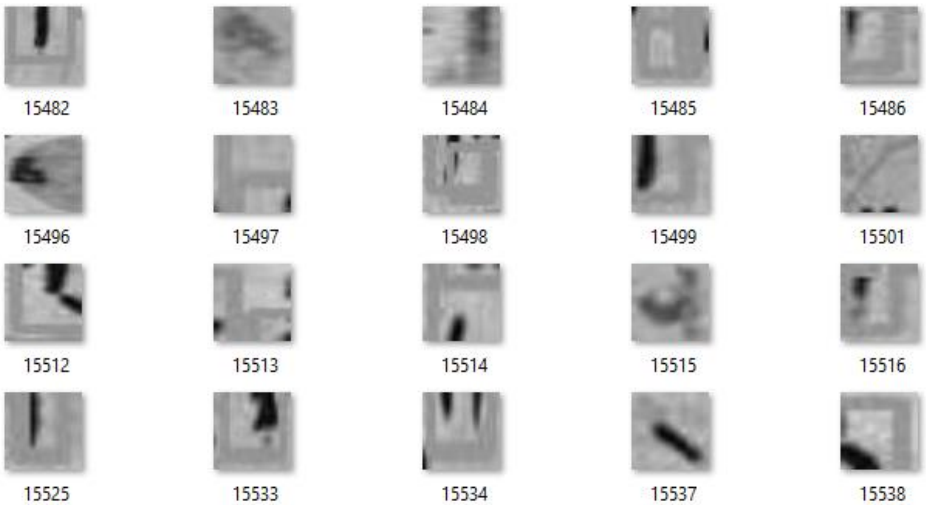



**Figura 2.27. Cluster cu părți de caractere.**

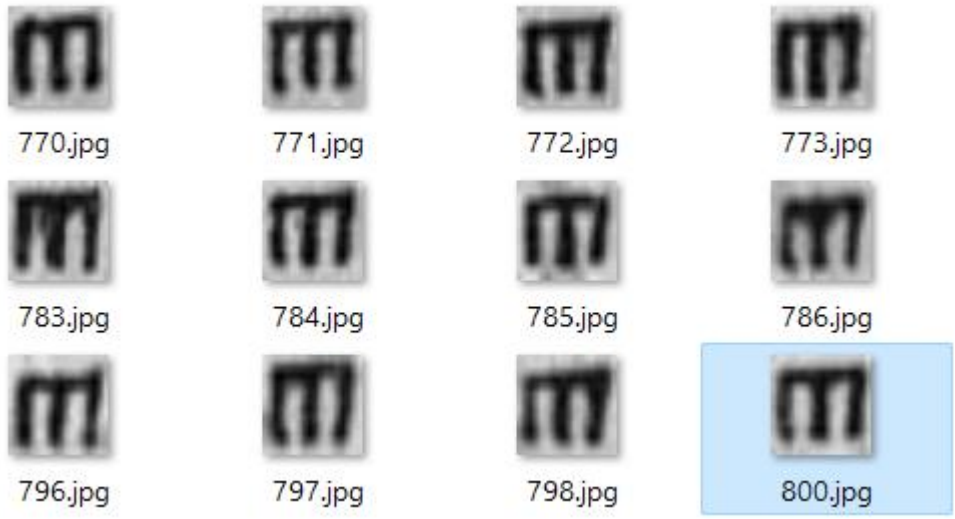



**Figura 2.28. Cluster format din glifele literei ‘m’ pentru fontul *B* (litera *t* în echivalent latin) din alfabetul chirilic utilizat în acest font.**

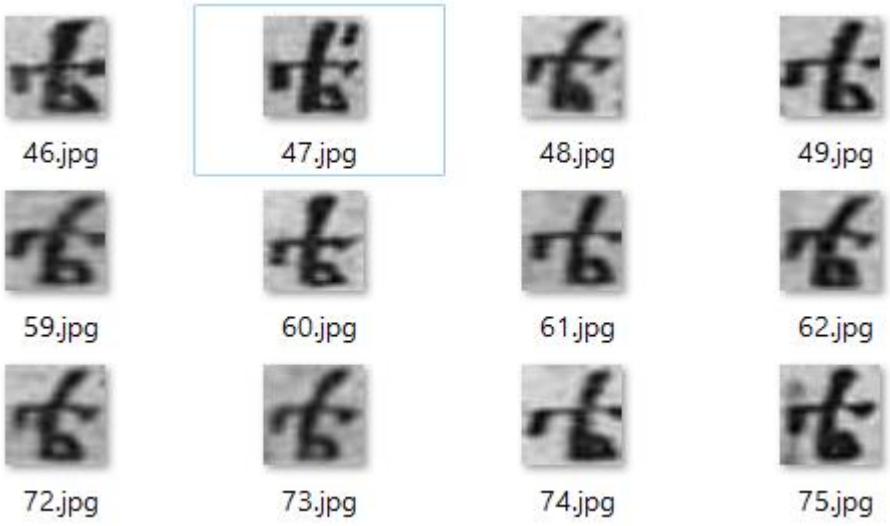



**Figura 2.29. Cluster format din glifele literei ‘ѣ’ pentru fontul *A* (echivalentul latin este combinația de litere ‘*ea*’).**

După curățarea și ajustarea manuală a clusterelor vom plasa imaginile din ele în două foldere, unul pentru setul cu fontul *A* și altul pentru caracterele cu fontul reprezentat de setul *B*. Rezultatul acestui proces sunt două dosare denumite *fontA* și *fontB*. În folderul *fontA* sunt peste 21 de mii de caractere, iar folderul *fontB* conține peste 8700. Poate părea un dezechilibru între numărul de exemple pentru fiecare dintre cele două fonturi, dar acest număr poate fi gestionat la etapa de împărțire a setului de date, dacă vom avea o acuratețe scăzută la clasificare.

Ultimul pas pentru a încheia cu formarea unui set de date bine organizat îl va constitui conversia setului de date în format *IDX*[41], format cunoscut și prin faptul, că este utilizat pentru prezentarea cunoscutei baze de date a cifrelor scrise de mână *MNIST*[42]. Vom face acest lucru folosind funcția *idx_converter()* din pachetul *idx_tools* care preia o structură de fișiere direct din sistemul de operare și o salvează în format IDX. Din moment ce dorim să antrenăm o rețea neurală, ar trebui să împărțim imaginile într-un set de date de antrenare și de testare. Pentru aceasta vom muta 30% din imaginile din fiecare set *A și B* într-un folder de testare. Ieșirea va fi constituită din 4 fișiere: 2 fișiere cu imagini pentru antrenare și testare și alte 2 fișiere cu etichetele corespunzătoare fișierelor cu imagini. Etichetele cu clasa caracterelor vor fi valorile **0** și **1**, unde **1** este clasa caracterelor din *fontA* și **0** - clasa caracterelor din *fontB.* În urma acestor operațiuni vom obține peste 21200 de exemple de antrenare și peste 9 mii de exemple de testare.

În continuare vom antrena o rețea neurală (RN) cu setul de date pregătit la această etapă.

***Antrenarea unei RN pentru clasificarea caracterelor și evaluarea acesteia***

Vom instrui o rețea neurală multistrat (RNM) pe setul de date pregătit la etapa anterioară pentru a clasifica caracterele în cele două fonturi diferite. Această abordare va urma exemplul de pe pagina web *Tensorflow*[43], doar că arhitectura modelului RNM va fi caracteristică unei clasificări binare.

În primul rând, trebuie să încărcăm datele pe care le-am salvat anterior în formatul de date IDX. L-am împărțit deja într-un set de date de antrenare și testare. Fiecare dintre aceste două seturi de date vine împreună cu un fișier cu etichete pe care îl vom încărca. În Figura 2.30 sunt prezentate câteva exemple aleatorii pentru a verifica dacă setul de date a fost salvat corespunzător.

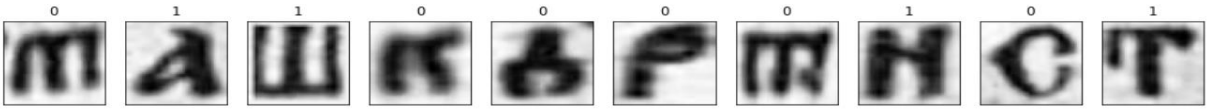



**Figura 2.30. Exemple aleatorii din setul de antrenare [30].**

Înainte de a începe să construim arhitectura RNM, trebuie să normalizăm toate imaginile, astfel încât valorile pixelilor să fie valori reale între 0 și 1 în loc de 0 până la 255. Pentru a face acest lucru vom raporta valorile pixelilor din imagini la 255.

[41] https://www.fon.hum.uva.nl/praat/manual/IDX_file_format.html
[42] http://yann.lecun.com/exdb/mnist/
[43] https://www.tensorflow.org/datasets/keras_example

Rețeaua neurală multistrat descrisă aici (vezi figura 2.31) este structurată în trei straturi principale: un strat de intrare, un strat ascuns și un strat de ieșire. Stratul de intrare este compus din neuroni care reprezintă fiecare o caracteristică distinctă $X_i$ a caracterului din imagine. În acest caz, există 2,500 de astfel de caracteristici reprezentate. Stratul ascuns al acestei rețele constă din 128 de neuroni și folosește funcția de activare *ReLU*[44] (Rectified Linear Unit). ReLU este o funcție de activare populară în rețelele neurale datorită faptului că nu atinge limita superioară în timpul propagării înapoi a erorii, ceea ce face antrenarea rețelei mai eficientă. Stratul de ieșire al rețelei conține un singur neuron, folosind o funcție sigmoidală de activare $\sigma$ definită ca

$$\sigma(x) = \frac{1}{1+e^{-x}} \quad (2.2)$$

unde $x$ este suma ponderată a ieșirilor $S_y$ din stratul ascuns. Funcțiile sigmoidale au formă de „S" și sunt adesea utilizate în rețelele neurale pentru o clasificare binară, deoarece produc un rezultat între 0 și 1, care poate fi interpretat ca o probabilitate.

Construirea rețelei neurale vom începe-o cu o transformare a datelor de intrare dintr-o matrice *x* pe *y* într-un vector de lungimea $x * y$ (în cazul nostru - 50*50). Acest lucru se face cu stratul *keras.layers.Flatten*[45]. Apoi vom adăuga 128 de neuroni în stratul ascuns care este complet conectat la ultimul strat. Un singur neuron în stratul de ieșire este suficient, deoarece avem o clasificare binară. Antrenarea rețelei neurale a fost realizată pe parcursul a 300 de epoci, durând aproximativ 55 de minute fără utilizarea unei unități de procesare grafică (GPU[46]).

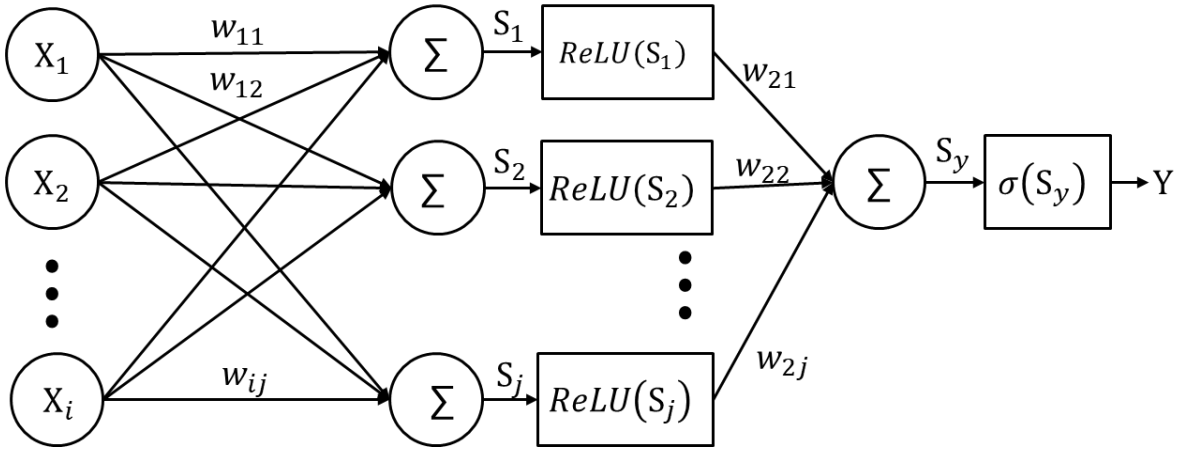



**Figura 2.31. Arhitectura rețelei neurale pentru clasificarea fonturilor.**

[44] https://keras.io/api/layers/activations/#relu-function
[45] https://keras.io/api/layers/reshaping_layers/flatten/
[46] https://en.wikipedia.org/wiki/Graphics_processing_unit

În urma antrenării constatăm o acuratețe de 96.7% (figura 2.31). Raportul prezicerilor la clasele actuale ale modelului RNM în matricea de confuzie este prezentat în figura 2.32. Dacă calculăm eroarea de clasificare utilizând datele fals-pozitive și fals-negative din matricea de confuzie, atunci avem $(128 + 174) * 100\%/9093 = 3.3\%$ eroare.

```
Epoch 295/300
21212/21212 [==============================] - 7s 348us/sample - loss: 0.0458 - accuracy: 0.9864
Epoch 296/300
21212/21212 [==============================] - 8s 357us/sample - loss: 0.0486 - accuracy: 0.9847
Epoch 297/300
21212/21212 [==============================] - 8s 359us/sample - loss: 0.0455 - accuracy: 0.9862
Epoch 298/300
21212/21212 [==============================] - 8s 364us/sample - loss: 0.0469 - accuracy: 0.9853
Epoch 299/300
21212/21212 [==============================] - 7s 353us/sample - loss: 0.0461 - accuracy: 0.9851
Epoch 300/300
21212/21212 [==============================] - 8s 366us/sample - loss: 0.0456 - accuracy: 0.9860
```

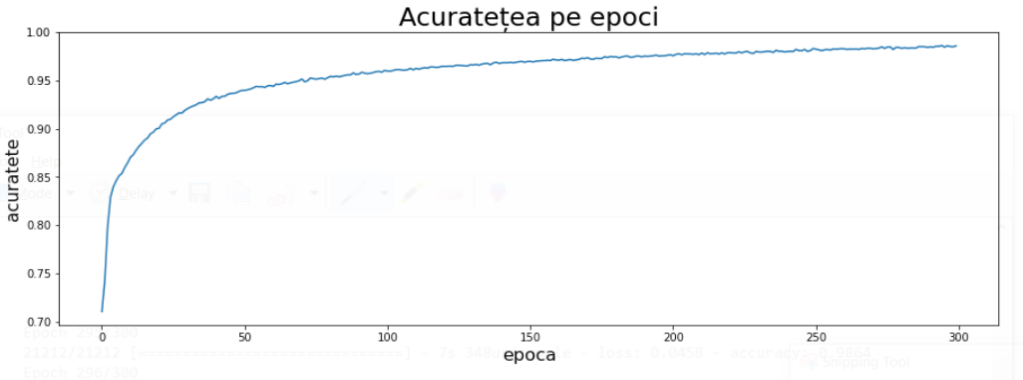



**Figura 2.32. Graficul cu istoria acurateței pentru fiecare epocă în parte.**

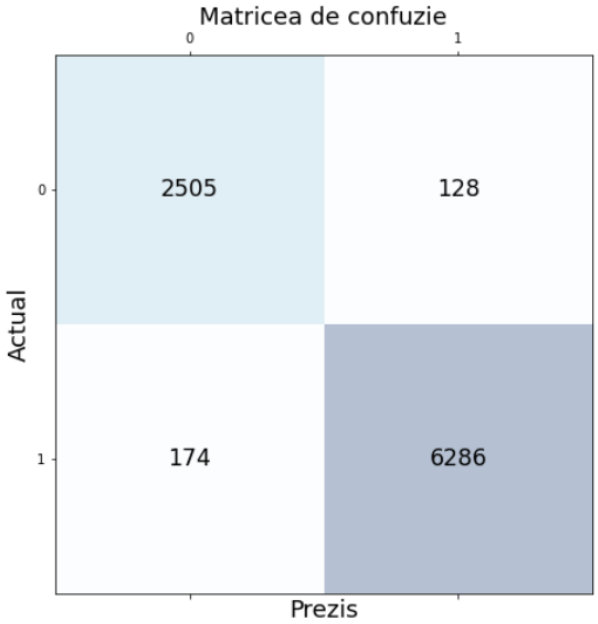



**Figura 2.33. Ilustrația matricei de confuzie pe setul de date de testare**.

Vom salva aparte structura modelului RNM într-un fișier *json* și ponderile rețelei într-un fișier HDF5[47]. La utilizarea modelului vom avea la intrare un set de caractere din pagini cu blocuri de texte în alfabet chirilic român tipărite în secolul XVII. Vom clasifica documentul pe baza clasei care a obținut numărul maximal de preziceri. După clasificarea documentului, FR 12 se va utiliza cu modelul OCR corespunzător fontului documentului.

## 2.8. Transliterarea din alfabetul chirilic român în alfabetul modern

Pentru a obține nu doar un document editabil, dar și unul ușor lizibil, de rând cu etapa de recunoaștere este necesară și cea de transliterare, deoarece alfabetele folosite la tipărirea textelor din diferite perioade istorice au trecut prin mai multe modificări. Parcursul istoric în acest aspect pentru limba română este prezentat în [120]. Transliterarea în acest sens este un tip de conversie a unui text dintr-un alfabet în altul care implică schimbarea literelor în moduri previzibile, cum ar fi pentru chirilică д → **d**, greacă χ → **ch/h**, armeană ն → **n**, latină æ → **ae**, sau chirilică românească veche ѧ → **î/în/îm** în dependență de context [121]. Prin urmare transliterarea constă în reprezentarea caracterelor unui alfabet *X* dat prin caracterele altui alfabet *Y*, păstrând (în măsura posibilităților) operația reversibilă.

Până în prezent, au fost propuse câteva tehnici de transliterare între două grafii. Un interes aparte au trezit lucrările orientate pe transliterarea ortografică a numelor proprii englezești în chineză, japoneză, coreeană sau arabă. În 1997, a fost introdusă o metodă de transliterare între japoneză și engleză, utilizând algoritmi de traducere bazați pe mașini cu stări finite, această metodă fiind adaptată în anul următor pentru transliterare bidirecțională între engleză și arabă [122].

În [123] se propune o tehnică de transliterare numită mapare ortografică directă (*direct orthographic mapping* sau DOM), cu alte cuvinte, un model de transliterare pe bază de n-grame. Abordarea DOM încearcă să modeleze asocierea echivalentă fonetică prin explorarea completă a informațiilor contextuale ortografice și a mapării ortografice. În tehnica DOM se prezintă un model comun de transliterare pentru a capta relația de mapare ortografică *sursă-țintă* și informațiile contextuale. Cadrul propus este aplicabil tuturor perechilor de limbi străine.

Procedura de transliterare se elaborează după sistemele de scriere ale limbilor care sunt supuse transliterării. Standardele pentru conversia sistemelor de scriere se orientează spre un sistem riguros, complet reversibil și univoc [124].

[47] https://www.h5py.org/

Transliterarea textelor de limbă română din scrierea „moldovenească” în cea latină în țara noastră a început în anul 1989 când a fost aplicată Legea nr. 3462 din 31 august 1989, adoptată de Parlamentul Republicii Moldova [125]. De exemplu, în tradiția limbii române litera chirilică “***х***” se transliterează prin litera latină “***h***”, dar în tradiția țărilor anglofone aceeași literă se transliterează prin “***kh***”. Astfel, în română token-ul “***Сахалин***” se translitерează “***Sahalin***”, iar în engleză ca “**Sakhalin**” [126].

În acest compartiment vom examina problemele, care apar la transliterarea textelor tipărite cu alfabetul chirilic român (ACR), utilizat în sec. XIV-XIX (cu anumite modificări).

Una din aceste probleme o constituie prezentarea textului chirilic recunoscut în calculator. De fapt, puține fonturi în prezent au codurile literelor din ACR. Dintre ele fac parte: Unifont și Everson Mono, care au apărut abia în anul 2009. Unele litere care folosesc anumite diacritice sau combinații de litere lipsesc definitiv și ar trebui puse în evidență în mod deosebit, de exemplu **ѹ̆** (ŭ) sau **іѹ̆**(iŭ). Pentru a fi prezentate în Unicode, este necesar să fie incluse și accentele, însă toate detaliile reprezentării grafice ale textului original este dificil de reprodus.

**Tabelul 2.2. Corespondența unor litere din ACR cu alfabetul modern al limbii române (AMR).**

| ACR | AMR | Codul Unicode pentru literele din ACR |
|---|---|---|
| “Ѣ” | “Ea” | 0462 |
| “ѣ” | “ea” | |
| “Ꙗ” | “Ia” | 0465 |
| “ꙗ” | “ia” | 0464 |
| “Ѧ” | “Î”, “În”, “Îm” | A64E |
| “ѧ” | “î”, “în”, “îm” | A65F |
| “Ꙋ” | “U” | A64A |
| “ꙋ” | “u” | A64B |

| | | |
|---|---|---|
| “Ҟ” | “C”, “Ch” (înainte de **e, i**) | 041A |
| “ҟ” | “c”, “ch” (înainte de **e, i**) | 043A |
| “Ъ” | “Ă” | 042A |
| “ъ” | “ă” | 044A |
| “Щ” | “Șt” | 0429 |
| “щ” | “șt” | 0449 |
| “Ҷ” | “Gi”, “Ge”, “G” (înainte de **e, i**) | 049F |
| “ҷ” | “gi”, “ge”, “g” (înainte de **e, i**) | 044F |

Conform cărții *Gramatica românească din 1797* de Radu Tempea, ACR conține 43 de litere, acestea fiind următoarele: Аа Бб Вв Гг Дд Єє Жж Ѕs Зз Ии Її Кк Лл Мм Нн Оо Пп Рр Сс Тт Ꙋꙋ Ꙋꙋ Фф Хх Ѡѡ Цц Чч Шш Щщ Ъъ Ыы Ьь Ѣѣ Ѫѫ Юю Ꙗꙗ Ѧѧ Ѳѳ Ѱѱ Ѯѯ Ѵѵ Ѧ̀ѧ̀ Ѫ̀ѫ̀.

Majoritatea literelor (37) sunt transliterate folosind reguli simple, independente de context (tabelul 2.3). Luând în considerare că litera “ъ” este un semn moale, ea va fi transliterată ca fiind un și de caractere gol “” (empty string[48]). De exemplu, “глас**ъ**” trece în “glas”.

**Tabelul 2.3. Reguli simple de transliterare din ACR în AMR.**

| **ACR -> AMR** | | **ACR -> AMR** | | **ACR -> AMR** | | **ACR -> AMR** | |
|---|---|---|---|---|---|---|---|
| “a” | “a” | “л” | “l” | “ф” | “f” | “ю” | “iu” |
| “б” | “b” | “м” | “m” | “х” | “h” | “ѧ” | “ia” |
| “в” | “v” | “н” | “n” | “ѡ” | “o” | “ѳ” | “t” |

[48] https://en.wikipedia.org/wiki/Empty_string

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| “д” | “d” | “о” | “о” | “ц” | “ţ” | “ψ” | “ps” |
| “е” | “е” | “п” | “р” | “ш” | “ş” | “ğ” | “х” |
| “ж” | “j” | “р” | “г” | “щ” | “şt” | “v” | “i” |
| “ѕ” | “dz” | “с” | “s” | “ъ” | “ă” | “кс” | х” |
| “з” | “z” | “т” | “t” | “ы” | “î” | | |
| “и” | “i” | “ɣ” | “u” | “ь” | “” | | |
| “ї” | “i” | “оу” | “u” | “ж” | “ă” | | |

Cele 6 litere rămase folosesc reguli compuse, în dependență de context (tabelul 2.4).

**Tabelul 2.4. Reguli de transliterare compuse din ACR în AMR.**

| **ACR -> AMR** | | **Context** |
|---|---|---|
| “г” | “gh” | înainte de **е, и, ї, ю** |
| “г” | “g” | în restul cazurilor |
| “к” | “ch” | înainte de **е, и, ї, ю** |
| “к” | “c” | în restul cazurilor |
| “ч” | “c” | înainte de **е, и, ѣ** |
| “ч” | “ce” | înainte de **а** |
| “ч” | “ci” | în restul cazurilor |
| “џ” | “g” | înainte de **е, и** |
| “џ” | “ge” | înainte de **а** |
| “џ” | “gi” | în restul cazurilor |
| “ѣ” | “e” | după **ч;**<br>excepție **чѣ -> cea** |

| | | |
|---|---|---|
| “ѣ” | “ea” | în restul cazurilor |
| “ѧ” | “a” | la începutul cuvântului;<br>după **ї, ц** |
| “ѧ” | “e” | după **ч** |
| “ѧ” | “ea” | după oricare consoană, la sfârșitul cuvântului |
| “ѧ” | “ia” | în restul cazurilor |
| “ѧ” | “î” | înainte de **н, м** |
| “ѧ” | “îm” | înainte de **б, п** |
| “ѧ” | “în” | în restul cazurilor |

## 2.9. Instrumentul software de transliterare din ACR în AMR

În mod formal, transliterarea este un sistem de reguli parametrizate care se aplică la fiecare al $\boldsymbol{n}$-lea caracter $\boldsymbol{x_n}$ al unui cuvânt *X*. Rezultatul $\boldsymbol{y_n} = Trans(x_n, Pos(\boldsymbol{n}, \boldsymbol{X}))$ reprezintă o secvență de caractere ale căror concatenare reproduce forma transliterată a cuvântului *Y*. În contextul dat, avem următoarele definiții pentru variabilele din formulă:

1. $y_n$: al *n*-lea caracter din cuvântul transliterat *Y*.
2. $x_n$: al *n*-lea caracter din cuvântul original *X*.
3. *X*: cuvântul original în sistemul de scriere sursă.
4. *Y*: cuvântul transliterat în sistemul de scriere țintă.
5. *n*: poziția curentă a caracterului în cuvânt (începând de la 0).
6. *Trans()*: funcție care aplică regulile de transliterare pentru un caracter dat ($x_n$) și poziția sa (*n*) în cuvântul *X*.
7. *Pos*(*n, X*): funcție care determină poziția caracterului $x_n$ în cuvântul *X*.

Există însă mai multe excepții de la această formulă simplă, acestea fiind constituite din cuvinte străine, nume proprii etc., care cer o abordare distinctă, spre exemplu, prin utilizarea unor dicționare cu astfel de cuvinte excepționale.

Partea de back-end a aplicației este scrisă în Java, folosind tehnologii compatibile cu toate caracterele din Unicode. Dacă fontul este înregistrat și instalat în sistemul de operare, metodele Java folosite pentru interfață soluționează problema de afișare a caracterelor cu o simplă referință la fontul în cauză.

Prima interfață grafică a utilității de transliterare este construită utilizând JavaFX și are un design potrivit unei aplicații desktop. Aplicația este compatibilă cu următoarele formate de fișiere: .doc, .docx, .rtf, .txt. Luând în considerare cerința de asigurare a unui acces cât mai larg spre acest serviciu, a fost elaborată și varianta Web a instrumentului de transliterare denumit *AAConv*[49], care constă numai din partea de front-end, conectată cu partea back-end dezvoltată anterior. Tehnologiile folosite la elaborarea aplicației Web sunt cele obișnuite, precum HTML, CSS, JavaScript. Fereastra principală a aplicației desktop este prezentată în Figura 2.33, iar în figura 2.34 - aplicația Web.

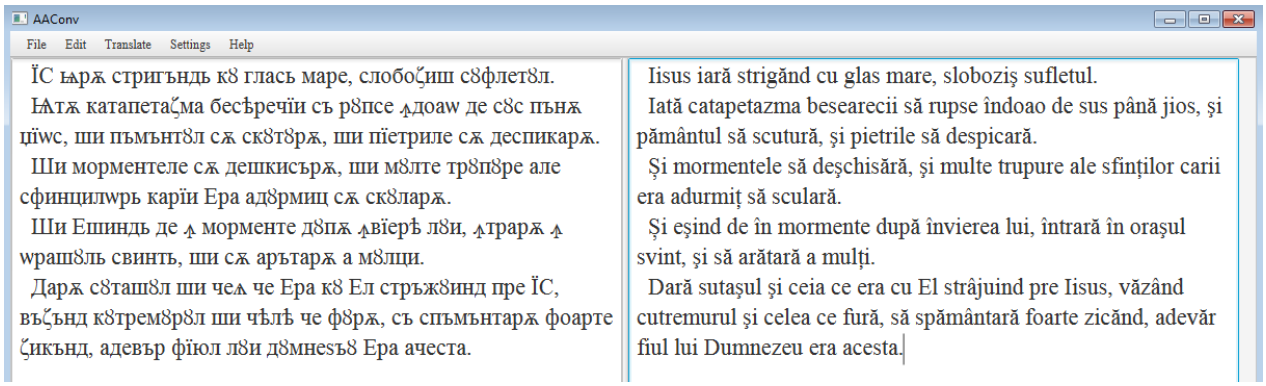



**Figura 2.34. Aplicația Desktop de transliterare „AAconv".**

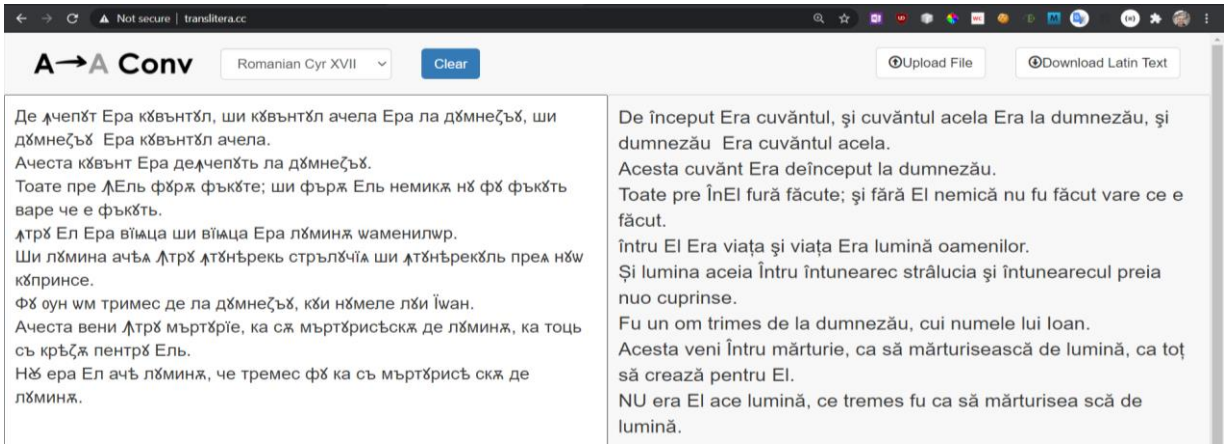



**Figura 2.35. Aplicația Web de transliterare din ACR în AMR.**

Ambele aplicații au opțiunea de a alege perioada textului ce urmează a fi transliterat, opțiunea de a actualiza/păstra scrierea cuvintelor. Aplicația desktop mai dispune și de opțiunea de a detecta automat perioada textului introdus pentru a fi transliterat. Precizia de transliterare din ACR în AMR este de aproximativ 95%.

[49] https://translitera.cc/

## 2.10. Instrumente de aliniere a textelor vechi la cele moderne

Alinierea unui text vechi la reprezentarea sa modernă înseamnă traducerea acestuia într-un limbaj contemporan prin înlocuirea variantelor lexicale învechite cu expresii moderne.

Textele paralele sunt resurse lingvistice valoroase în multe domenii de cercetare, dar și în aplicații practice. Cea mai cunoscută aplicație este *traducerea automată statistică* sau, mai recent, *traducerea automată neurală* (folosind rețele neurale). De asemenea, texte paralele se studiază în contextul dezambiguizării sensului cuvântului, recunoașterii numelor proprii, învățării modelului lingvistic, dar și în analiza diacronică a limbajelor naturale.

Un set de texte paralele poate forma un *corpus paralel*. Partea de bază în lucrul cu un corpus paralel este sarcina de *aliniere*. Alinierea în acest sens este procesul de identificare și legare a părților textuale corespunzătoare din textele paralele. O caracteristică importantă a textelor paralele este proprietatea de a avea în sine o corespondență între două sau mai multe texte, de exemplu, echivalența traducerii sau parafrazării. Noi presupunem că această echivalență o constituie *sensul*.

În lingvistica computațională, termenul *text paralel*, se referă și la perechi de seturi de texte din același domeniu, care includ traduceri ale aceluiași document. Dar, în majoritatea cazurilor, textele paralele se referă la corpusuri bilingve.

Un *corpus paralel diacronic* este un corpus paralel în care textele paralele au fost scrise în aceeași limbă, dar în diferite perioade de timp, spre exemplu textul din *Noul Testament din sec. XVII* în paralel cu textul din *Noul Testament publicat la sfârşitul sec. XX*.

Scopul cercetării noastre din acest subcapitol este de a transforma textele istorice în texte care folosesc expresii și cuvinte din dicționarul modern al limbii române. Începutul lucrărilor este elaborarea unui corpus paralel diacronic cu text în limba română scris cu peste 350 de ani în urmă, aliniat la varianta sa modernă. Sursele noastre primare sunt: textul din *Noul Testament tipărit în 1648 la Bălgrad* și o variantă electronică modernă a acestui document din 1990.

### Corpusul diacronic paralel

La prima etapă de *aliniere* am construit un corpus paralel diacronic (în continuare *CPD*[50]), plasat în Internet cu acces deschis [127, 128]. S-ar putea spune că această resursă nu poate fi numită încă corpus paralel, ci doar text paralel. Se lucrează asupra extinderii acestei resurse digitalizând și adăugând texte, astfel încât să obținem un corpus paralel diacronic veritabil.

[50] https://github.com/bumbutudor/TextAlignUtils/

CPD conține textul din cartea „*Noulu Testament sau Înpacarea, au Leagea noao a lui Is. Hs.*” (în continuare NTV), tipărit în 1648 la Cetatea Bălgradului, Transilvania și textul din *versiunea electronică a Noului Testament* (în continuare NTM, pregătit și adnotat de Bartolomeu Valeriu Anania, arhiepiscop al Arhiepiscopiei Vadului, Feleacului și Clujului, în 1990). Volumul de text din CPD exprimat prin propoziții este de circa 8400 de propoziții (câte ~4200 de propoziții din fiecare perioadă în parte) [129].

Luând în considerare că alfabetul cărții tipărite în secolul al XVII-lea este alfabetul chirilic român, adică unul vechi, este nevoie de mai mulți pași pentru a ajunge la textul editabil cu alfabet modern al limbii române.

La primul pas este aplicat un șablon OCR antrenat pe texte din sec. XVII, unde am obținut un rezultat cu acuratețea de peste 75%, iar erorile au fost corectate manual. Cauzele unei precizii mediocre sunt bine cunoscute, fiind relatate mai sus, dar unele le vom evidenția repetat, și anume: *paginile din carte au pete maronii* și *unele rânduri de text sunt subliniate cu cerneală roșie*; *se întâlnesc des ligaturi verticale și cuvinte scrise împreună*; *referințele sunt scrise în slavonă cu un font mai mic și cu multe abrevieri*; *utilizarea masivă a accentelor* (Figura 2.35).

**Figura 2.36. Două fragmente din NTV [119].**

După pasul OCR are loc transliterarea din ACR în AMR. La această etapă folosim aplicația Web de transliterare AAConv. Un exemplu de text din NTV în CPD este prezentat în Tabelul 2.5. Acest text este recunoscut și transliterat, iar ambele variante se păstrează în corpus. Acuratețea obținută la transliterare se plasează în intervalul 93-98% luând în considerare problemele regulilor complexe. În continuare vom încerca să aliniem în CPD textul din NTV transliterat cu textul din NTM.

**Tabelul 2.5. Fragment de text recunoscut și transliterat din CDP [119].**

| **Textul după OCR** | **Textul după transliterare** |
|---|---|
| Де ѧчепѹт Ера кѹвънтѹл, ши кѹвънтѹл ачела Ера ла дѹмнеζъѹ, ши дѹмнеζъѹ Ера кѹвънтѹл ачела. | De început Era cuvântul, și cuvântul acela Era la dumnezău, și dumnezău Era cuvântul acela. |
| Ачеста кѹвънт Ера деѧчепѹть ла дѹмнеζъѹ. | Acesta cuvânt Era deînceput la dumnezău. |
| Тоате пре ѧЕль фѹрѫ фъкѹте; ши фърѫ Ель немикѫ нѹ фѹ фъкѹть варе че е фъкѹть. | Toate pre ÎNEl fura făcute; şi fără El nemica nu fu făcut vare ce e făcut. |
| ѧтрѹ Ел Ера вїѩца ши вїѩца Ера лѹминѫ wаменилwр. | îtru El Era viața şi viața Era lumina oamenilor. |
| Ши лѹмина ачѣѧ ѧтрѹ ѧтѹнѣрекь стрълѹчїѧ ши ѧтѹнѣрекѹль преѧ нѹw кѹпринсе. | Și lumina aceaia Întru întunearec strălucia şi întunearecul preia nuo cuprinse. |
| Фѹ оун wм тримес де ла дѹмнеζъѹ, кѹи нѹмеле лѹи Їwан. | Fu un om trimes de la dumnezău, cui numele lui Ioan. |
| Ачеста вени ѧтрѹ мъртѹрїе, ка сѫ мъртѹрисѣскѫ де лѹминѫ, ка тоць съ крѣζѫ пентрѹ Ель. | Acesta veni Întru mărturie, ca sa mărturiseasca de lumina, ca toț să creaza pentru El. |
| Нѽ ера Ел ачѣ лѹминѫ, че тремес фѹ ка съ мъртѹрисѣ скѫ де лѹминѫ. | NU era El acea lumina, ce tremes fu ca să mărturisea sca de lumina. |
| Ачеста Ера лѹмина чѣ адевъратѫ карѣ лѹминѣζѫ пре тот wмѹл, чела че вине ѧлѹме. | Acesta Era lumina cea adevărata carea lumineaza pre tot omul, cela ce vine înlume. |
| ѧ лѹме Ера ши лѹмѣ прен Ель съ фъкѹ, ши лѹмѣ Пре Ель нѹ кѹноскѹ. | În lume Era şi lumea pren El să făcu, şi lumea Pre El nu cunoscu. |
| ѧтрѹ лѹи вени, ши аи лѹи пре Ель нѹ прїимирѫ. | Întru lui veni, şi ai lui pre El nu priimira. |

La următoarea etapă ne vom ocupa de punerea în corespondență a versetelor din NTV cu cele din NTM. Aici, am luat aproximativ 3600 de versete din fiecare text în parte. Versetele au fost selectate în funcție de ordinea lor din carte, respectiv: 985 versete din *Evanghelia după Matei* (începând cu capitolul 4), 671 de versete din *Evanghelia după Marcu*, 1130 de versete din

*Evanghelia după Luca* și 873 din *Evanghelia după Ioan*. Structura și lexicul aceleiași propoziții în resursa veche și modernă variază (vezi Tabelul 2.6).

**Tabelul 2.6. Eșantion de versete din *Evanghelia după Marcu* aliniate în CPD.**

| **Nr.** | **Versete NTV** | **Versete NTM** |
|---|---|---|
| 1. | Și fu îzilele acelia, veni Iisus de în nazareful galileei, şi să boteză dela Ioan în Iordan. | Și în zilele acelea, Iisus a venit din Nazaretul Galileii și S'a botezat în Iordan de către Ioan. |
| 2. | Și aciiș eșind de în apă, văzu deşchise ceriurele, şi duhul ca un porumbu, pogorănd spre El. | Și îndata, ieșind din apă, a vazut cerurile deschise și Duhul ca un porumbel pogorându-Se peste El. |
| 3. | Și glas fu den ceriure, tu ești fiul mieu cel iubit, întru carele bine voescu. | Și glas s'a făcut din ceruri: "Tu ești Fiul Meu Cel iubit, întru Tine am binevoit". |
| 4. | Și numai decăt scoase pre El duhul împustie. | Și îndata Duhul L-a scos în pustie. |
| 5. | Și era acolo împustie, patruzeci de zile şi patruzeci de nopți, ispitiț de satana: şi era cu fierile, şi îngerii slujia lui. | Și a fost în pustie patruzeci de zile, fiind ispitit de Satana. Și era împreuna cu fiarele, și îngerii Îi slujeau. |
| 6. | Iară după prinsoarea lui Ioan, veni Iisus îngalilea, propoveduind Evanghelie a Împărăției lui dumnezău. | După ce Ioan a fost întemnițat, Iisus a venit în Galileea, propovăduind Evanghelia împărăției lui Dumnezeu |

Pentru a măsura similitudinea dintre versete, folosim potrivirea fuzzy a secvențelor pe baza distanței *Levenstein* din pachetul *fuzzywuzzy*[51] din Python. Raportul de similaritate este măsurat în procente. Am comparat diferite combinații pentru a obține cel mai bun raport pentru textul nostru.

Prima combinație este *raportul simplu* (simple ratio) care compară întreaga similitudine a șirurilor de caractere, în ordine. Folosind perechea de versete nr. 2 din Tabelul 2.6, am obținut o similaritate de 68%. A doua combinație este *raportul tokenurilor sortați* (token sort ratio) care compară întreaga similitudine a secvenței, dar ignoră ordinea cuvintelor în versete. Aici am obținut 68% de similaritate pe aceeași pereche de versete. Cea de-a treia combinație din experimentul nostru este *raportul setului de tokenuri* (token set ratio) care compară întreaga similitudine a secvenței, dar ignoră ordinea cuvintelor și duplicatele. Raportul pe baza *setului de tokenuri* ne

[51] https://pypi.org/project/fuzzywuzzy/

oferă rezultatul de 71% de similitudine dintre versetele din perechea nr. 2. Cu toate că această combinație are avantajul complexității (*O(n)*), ordinea cuvintelor și păstrarea cuvintelor care se repetă în verset sunt factori mult mai importanți pentru noi.

Conform rezultatelor obținute, am decis să aplicăm modelul *raport simplu* pentru a alinia propozițiile care depășesc pragul de 70% de similitudine. Prin urmare toate versetele au fost împărțite în propoziții (un verset poate include una sau mai multe propoziții). Listele de propoziții din NTV, NTM au fost comasate într-o singură listă împreună cu raportul de similitudine dintre ele în formă de triplete *(ntv_prop, ntm_prop, raport)* prezente în Figura 2.36.

```
('Acesta cuvânt Era deînceput la dumnezău.', 'Acesta era întru început la Dumnezeu.', 73)
('întru El Era viața și viața Era lumina oamenilor.', 'Întru El era viață și viața era lumina oamenilor.', 89)
('Fu un om trimes de la dumnezău, cui numele lui Ioan.', 'Fost-a om trimis de la Dumnezeu, numele lui era Ioan.', 78)
('Acesta veni Întru mărturie, ca sa mărturiseasca de lumina, ca toț să creaza pentru El.', 'Acesta a venit spre mărturie,
ca să mărturisească despre Lumină, ca toți să creadă prin el.', 79)
```

**Figura 2.37. Un fragment cu triplete *(ntv_prop, ntm_prop, raport)* unde *ntv_prop* este o propoziție din NTV, *ntm_prop* este o propoziție din NTM, iar *raport* este raportul simplu de similitudine dintre *ntv_prop* și *ntm_prop.***

În total am obținut peste 8 mii de propoziții (nu versete!) aliniate în CPD. Într-un final CPD include: *versetele din NTV* (cu varianta lor recunoscută și varianta transliterată); *versetele din NTM;* și *propozițiile NTV-VTM aliniate.* În secțiunea următoare vom prezenta proiectul elaborării *unui instrumentar de aliniere a textului vechi la text modern*.

### Instrumente de aliniere

Pentru a ajunge la alinierea automată a textelor vechi la textele moderne, mai sunt încă un număr semnificativ de pași de realizat. Un pas intermediar dintre acestea este *alinierea cuvintelor* vechi la cele moderne. Instrumentele de aliniere a cuvintelor sunt programe software care ajută un expert să pună în corespundere cuvintele din textul sursă cuvintelor din textul țintă. Vom examina câteva din cele existente. Unul dintre aceste instrumente este *Berkeley Word Aligner*[52] (BWA), un program scris în Java, utilizat în multe cazuri pentru traducerea automată.

BWA aliniază cuvintele într-un corpus paralel racordat la nivel de propoziții, folosind un model Markov Ascuns (HMM). Pentru a instrui un model de aliniere *HMM* este necesar și un al treilea text paralel pentru fiecare pereche de texte aliniate. Toate textele trebuie adnotate cu

[52] https://github.com/mhajiloo/berkeleyaligner

informații de analiză sintactică. La această etapă încă nu dispunem de un al treilea text paralel, prin urmare, instrumentul respectiv va fi utilizat mai târziu.

Un alt pachet de instrumente pentru *alinierea cuvintelor* este *GIZA ++*[53]. Acest instrument folosește, de asemenea, pe scară largă modelele ascunse HMM pentru a alinia textele. Giza ++ funcționează direct cu propozițiile aliniate din două texte paralele, fără a solicita textul marcat cu informații lingvistice suplimentare. Folosind un algoritm de maximizare a așteptărilor, rezultatele alinierii cuvintelor finale pot fi obținute după ce software-ul se antrenează cu un corpus paralel, prin mai multe iterații, de la textul sursă la textul țintă și invers. Acest pachet de instrumente rămâne în lista noastră de instrumente pentru a-l utiliza.

Luând în considerare că obiectul nostru de studiu sunt texte paralele diacronice, am decis să creăm un instrument de aliniere care să satisfacă nevoile noastre. Câteva exemple de necesitate ar fi: *calcularea scorului BLEU între texte, propoziții, expresii, cuvinte*; *vizualizarea interactivă de n-grame*; *vizualizarea arborelui de acoperire a cuvântului/expresiei*; *lucrul cu mai mult de două texte paralele în același timp etc*.

Prin urmare, avem nevoie de un editor de aliniere a cuvintelor/expresiilor paralele diacronice propriu, arhitectura căruia vom descrie-o în continuare. Acesta constituie o aplicație WEB, proiectată în cadrul Django din Python. Aplicația este formată din 3 module generale: *modulul de editare a textului paralel și modulul de formare a corpusului paralel*; *modulul de procesare a textului* și *modulul de învățare automată*.

Modulul de procesare a textului reprezintă punctul central în aplicația de aliniere a textelor vechi la textele moderne. Acesta permite vizualizarea și editarea simultană a mai multor texte paralele, asigurând propagarea automată a modificărilor în corpusurile paralele. Funcționalitățile-cheie ale modulului de procesare a textului includ: calcularea, numărarea și vizualizarea N-gramelor la nivel de token-uri (conform Figurii 2.37); atribuirea identificatorilor numerici (ID) tokenurilor, astfel încât fiecărui cuvânt să îi corespundă un număr întreg unic; calcularea scorului *BLEU*[54] pentru evaluarea similarității între propoziții și texte.

Pe lângă aceste funcționalități, arhitectura preconizată va conține și o componentă de adnotare a textului folosind metodologia *Punctilog* descrisă în [130-131]. Această componentă va fi focalizată pe operația de asociere pentru dezambiguizarea textelor paralele și maparea tuturor constituenților. Punctilog va permite identificarea și reprezentarea clară a structurii gramaticale și a relațiilor dintre cuvinte în ambele versiuni ale textului, facilitând alinierea textelor vechi la cele moderne.

[53] https://github.com/moses-smt/giza-pp
[54] https://en.wikipedia.org/wiki/BLEU

Modulul de învățare automată implică selectarea/configurarea unor arhitecturi de rețele neurale speciale care să învețe din corpusurile paralele să alinieze textele. Acest modul nu are funcționalități complet diferite, dar ne concentrăm pe un dispozitiv puternic de vizualizare și interactivitate.

Una dintre funcționalitățile de bază pe care le vom folosi pentru a evalua și îmbunătăți similitudinea dintre textele paralele diacronice este scorul *BLEU*, o valoare utilizată pentru a evalua o propoziție generată în traducerea automată în comparație cu o propoziție de referință (tradusă de un expert) [132]. Această metodă are multe avantaje: este rapidă și ușor de calculat; se corelează foarte mult cu evaluarea umană a traducerii. Ideea acestei abordări este de a număra N-gramele în textul tradus și în textul de referință, secvențele cărora se potrivesc, unde un *uni-gram* ar fi un cuvânt și un *bi-gram* ar fi fiecare pereche de cuvinte. Comparația se face indiferent de ordinea cuvintelor și poate ajunge până la *4-grame*. Vom sublinia că în corpusul nostru (CPS) *textul sursă* este reprezentat de NTV, iar textul de referință este textul din NTM.

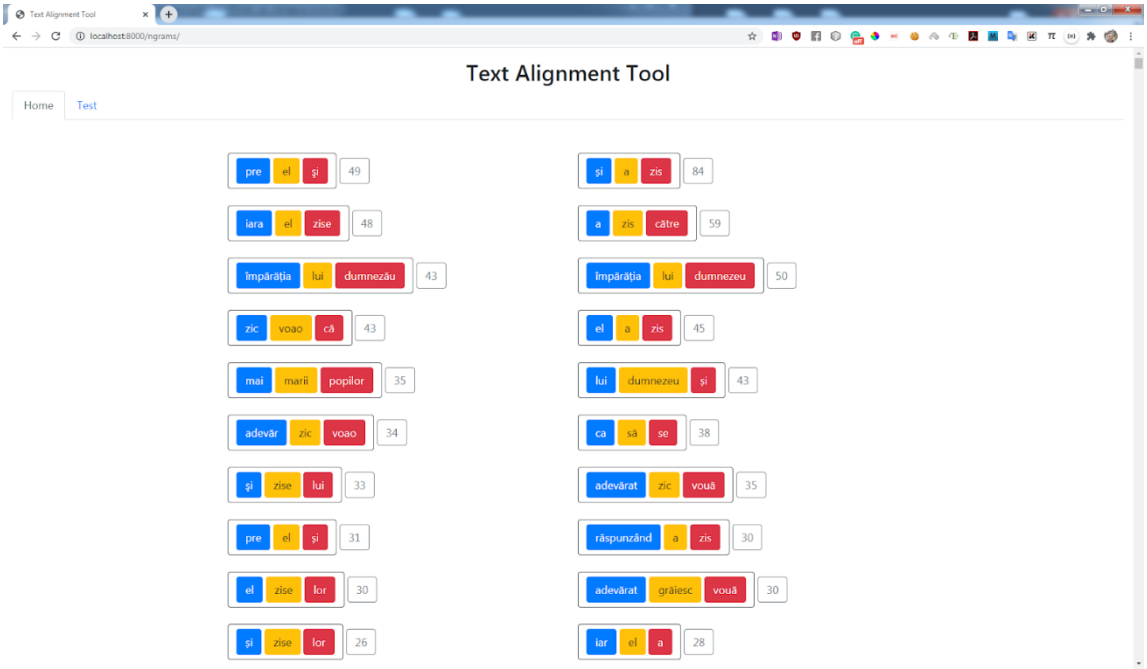



**Figura 2.38. Aplicația Web de aliniere a textelor.**

Rezumând acest compartiment vom menționa existența diferitelor abordări pentru alinierea cuvintelor într-un text paralel. Din cele trei instrumente descrise mai sus cel dezvoltat de noi este orientat la încadrarea unor funcționalități speciale necesare pentru a alinia cuvintele într-un corpus paralel diacronic. O funcționalitate specifică care este integrată implicit în aplicația noastră este calcularea scorului BLEU între propoziții.

## 2.11. Concluzii la capitolul II

Utilizând în calitate de instrument de bază sistemul software FR15, a fost stabilit că componentele acestuia pot fi extinse și adaptate și pentru cazul recunoașterii tipăriturilor vechi românești (în cazul nostru – din sec. XVII), care nu erau prevăzute în softul inițial. Evaluarea procedurii de învățare implicând mai multe pagini de antrenare și testare în cadrul unui proces iterativ a demonstrat că odată cu creșterea numărului datelor de antrenare acuratețea modelului crește semnificativ, atingând valori acceptabile (0.96 în cazul operării cu dicționar și 0.95 în cazul operării fără dicționar) la nivel de recunoaștere corectă a caracterelor chiar și după un număr nu prea mare de pagini (5-7 pagini). La nivel de cuvinte valoarea acurateței este mai mică, fapt ce denotă necesitatea utilizării unui număr mai mare de pagini pentru instruire. O problemă comună întâlnită în cărțile tipărite în secolul XVII o constituie scrierea unor litere deasupra altor litere sau folosirea abrevierilor cu tilde sau alte semne diacritice, inclusiv cifre scrise cu litere [133]. Pentru a obține o mai bună acuratețe a modelelor OCR pentru documentele chirilice românești tipărite în secolul XVII, rezoluția imaginilor trebuie să fie ridicată, deoarece acestea conțin multe detalii.

Putem spune că cea mai utilizată metodă de adăugare a dicționarelor în FR este prin crearea dicționarului din textul documentului care a fost recunoscut, dar este important să ținem cont și de lărgirea continuă a dicționarelor prin transliterarea vocabularelor existente din alfabetul modern în cel chirilic sau adăugarea cuvintelor în dicționar prin includerea cuvântului subliniat cu linie roșie în dicționar în timpul procesului de verificare ortografică.

Procesarea imaginilor din documentele vechi poate fi realizată cu softuri existente, dar și în acest caz nu avem la dispoziție unul, care ar satisface tuturor cerințelor noastre și este necesar de îmbinat funcționalitățile mai multor din ele. Astfel constatăm, spre exemplu, că nu este suficient să folosim doar modulele de preprocesare din FR deoarece acesta nu este dotat cu funcția de îngroșare a caracterelor. Pe de altă parte, Scan Tailor oferă un modul special pentru gestionarea grosimii liniilor. De asemenea, Scan Tailor oferă o metodă îmbunătățită de binarizare utilizând netezirea Savitzky-Golay și eliminarea marginilor întrerupte. Prin eliminarea marginilor întrerupte se pot localiza și îndepărta cu succes o mulțime de linii întrerupte (pixeli cu zgomot), ceea ce prin FR nu se poate de realizat. Prin urmare, în tehnologia procesării textelor vechi trebuie să avem prevăzută posibilitatea combinării acestor două softuri.

Putem constata o acuratețe bună demonstrată și de algoritmul de clasificare a fonturilor. În urma antrenării rețelei neurale multistrat se obține o acuratețe de peste 96%. O arhitectură de rețea neurală convoluțională în combinație cu o rețea neurală recurentă ar putea îmbunătăți semnificativ această acuratețe luând-se în considerare ordinea caracterelor în text. În ceea ce privește

identificarea fontului utilizat într-un document tipărit în această perioadă, s-au propus și alte soluții, pe lângă identificarea automată utilizând rețele neurale. Una dintre ele constă în recunoașterea unei mostre din documentul respectiv cu toate modele OCR, urmată de alegerea modelului care oferă cea mai mare acuratețe, iar o altă soluție constă în clasificarea modelelor OCR după tipografii, ceea ce implică faptul că utilizatorul cunoaște tipografia corespunzătoare documentului. Ultima soluție a fost implementată într-un instrument numit „Model Selector", care permite utilizatorului să aleagă secolul și regiunea unde a fost tipărit documentul, urmată de alegerea tipografiei corespunzătoare.

La transliterarea din alfabetul chirilic român în alfabetul modern vom accentua regulile de transliterare a unor litere din ACR dependente de context (litere vecine). Cea mai problematică literă este "ѧ" care poate fi transliterată ca și "a", "e", "ea", "ia". Cu toate că am găsit unele reguli de dependențe cu anumiți vecini, totuși sunt cazuri de excepție în care transliterarea acesteia iese din tipare. De exemplu: "чѣѧ" și "кърꙋѧ" se vor translitera în ambele cazuri cu aceeași regulă ("ѧ" => "ia"), deși cuvântul "чѣѧ" poate trece în "ceia" și în "ceea". În mod ideal am avea nevoie și de reguli de dependență la nivel de cuvinte vecine. Cu toate acestea acuratețea de transliterare întrece 98%.

Referitor la sarcina de aliniere a textelor vechi la cele moderne putem spune că bazându-ne pe scorul *BLEU* putem evalua și îmbunătăți similitudinea dintre textele paralele diacronice într-o serie de repetări de înlocuire a expresiilor învechite în expresii moderne astfel găsind majoritatea expresiilor echivalente. Implementarea metodei Punctilog [130-131] va putea îmbunătăți eficiența procesului de aliniere, contribuind la o mai bună înțelegere a evoluției limbajului și la conservarea patrimoniului lingvistic. În acest mod a și început crearea unui corpus paralel diacronic cu texte românești din secolul XVII puse în corespondență cu texte echivalente din secolul XX.

# 3. PLATFORMĂ DE DIGITIZARE

În acest capitol vom vorbi despre o platformă web, numită și platformă de digitizare, care oferă acces la instrumente și resurse informaționale pentru digitizarea documentelor în limba română tipărite în grafie chirilică [134-135]. Platforma de digitizare reprezintă principalul rezultat aplicativ al acestei teze și include o aplicație de digitizare care are o interfață grafică interactivă bazată pe React[55] și un set de API-uri create și gestionate prin Django[56] și Django Rest Framework[57], care leagă interfața grafică cu utilizatorul de instrumentele și resursele pentru digitizarea documentelor chirilice românești. Această aplicație este construită din șapte pași consecutivi pentru a satisface nevoile de digitizare a documentelor chirilice românești, dar utilizatorii pot crea propriile aplicații de digitizare folosind instrumentele și resursele disponibile pe platformă sau adăugând altele noi, cum ar fi modele OCR noi, module de prelucrare a imaginilor sau dicționare de cuvinte. Interfața grafică a acestei aplicații extinde o componentă *Stepper*[58] care afișează progresul prin pași numerotați oferind un flux de lucru asemănător unui *wizard*[59]. Aplicația de digitizare permite recunoașterea documentelor chirilice românești din secolele XVII-XX, transliterarea textelor în scrierea latină, editarea textelor recunoscute/transliterate, precum și descărcarea sau publicarea rezultatelor.

Scopul platformei de digitizare este de a oferi acces la instrumente și resurse informaționale utilizate pentru a procesa documentele din patrimoniul istoric românesc menționat în capitolul II. Instrumentarul include: motoare de preprocesare/prelucrare a imaginilor, cum ar fi Scan Tailor, ABBYY FineReader și OpenCV; modele de recunoaștere optică a caracterelor chirilice, care au fost antrenate pe seturi de date colectate din documente tipărite în secolele XVII, XVIII, XIX și XX; o aplicație de transliterare din grafia chirilică în cea latină; tastaturi virtuale, specifice alfabetelor folosite în perioadele menționate mai sus, precum ar fi alfabetul chirilic românesc, alfabetul de tranziție, alfabetul chirilic sovietic. Integrarea tuturor acestor instrumente într-o singură platformă oferă utilizatorilor un "ghișeu unic" – adică o interfață grafică unică pentru a controla și ghida procesul de digitizare al documentelor.

[55] https://reactjs.org/
[56] https://www.djangoproject.com/
[57] https://www.django-rest-framework.org/
[58] https://mui.com/material-ui/react-stepper/
[59] https://uxplanet.org/wizard-design-pattern-8c86e14f2a38/

### 3.1. Arhitectura platformei de digitizare

Platforma de digitizare conține un set de module pentru a facilita următoarele sarcini: procesarea imaginilor, recunoașterea documentelor și transliterarea textului, salvarea și publicarea rezultatelor. Aceste module sunt organizate în unități sau grupuri funcționale. Există și module comune care n-au fost incluse explicit în grupurile funcționale.

Primul grup funcțional se ocupă de prelucrarea imaginilor. Acest grup este dotat cu două module de preprocesare de bază, binarizarea imaginii și corectarea rezoluției, fiind extins cu module adiționale prin integrarea cu aplicații terțe, cum ar fi Scan Tailor, FineReader, OpenCV, GIMP etc.

Al doilea grup funcțional conține module care se ocupă de recunoașterea optică a documentelor. Acest grup include: selectarea modelului OCR în dependență de perioada istorică; folosirea unui dicționar de cuvinte la recunoaștere; editarea textului recunoscut, folosirea unui dicționar de excepții OCR, etc. Motorul OCR se bazează pe FineReader 15 și are modele antrenate cu seturi de date colectate din documente tipărite în secolele XVII, XVIII, XIX și XX utilizând motoarele de învățare din FineReader 15 și FineReader 12. Utilizatorii pot adăuga, de asemenea, modele OCR noi. Aceste modele au formatul FineReader XML (fișiere *.fbt*) care conțin configurațiile modelului OCR, setul de date de antrenare, alfabetul necesar, precum și dicționare de cuvinte.

Al treilea grup funcțional conține componentele ce se referă la transliterarea textului recunoscut. De rând cu transliterarea propriu-zisă, modulele din acest grup asigură actualizarea ortografiei (la solicitare), utilizarea dicționarelor de excepții pentru transliterare, editarea textului obținut în urma transliterării cu ajutorul unui verificator ortografic, corectarea automată a textului cu ajutorul unui sistem de inteligență artificială și, de asemenea, module secundare, precum: înlocuirea în text a apostrofului cu cratimă sau ștergerea din text a cratimelor care împart cuvintele la trecerea dintr-un rând în altul.

Al patrulea grup funcțional conține module destinate gestionării și publicării documentelor digitizate. Acesta include: salvarea textelor recunoscute/transliterate în diferite formate; descărcarea imaginilor prelucrate; încărcarea documentelor originale și a imaginilor prelucrate în cloud; salvarea stării obiectului digitizat în baza de date; extinderea dicționarelor de excepții în baza textelor obținute. Un modul special din acest grup îl constituie publicarea documentului digitizat. Publicarea documentelor digitizate poate fi un proces complex, deoarece trebuie să se ia în considerare mai mulți factori, cum ar fi drepturile de autor, conservarea documentelor și asigurarea accesului public. Pentru aceasta se prevede dezvoltarea unui modul de verificare și

aprobare pentru publicare. De asemenea, vor fi prevăzute mijloace pentru păstrarea și conservarea documentelor prin utilizarea unui depozit de arhivare digitală care poate oferi stocare pe termen lung, asigurând, după caz, și accesul liber către acestea. Documentul ar putea fi publicat pe platforme web sau biblioteci digitale care păstrează tezaurul digital, precum ar fi e-Moldova, Moldavica și Europeana, sau alte platforme solicitate de utilizator.

În figura 3.1 prezentăm arhitectura platformei de digitizare. Arhitectura cuprinde 4 grupuri funcționale (*G1–G4* în figura 3.1).

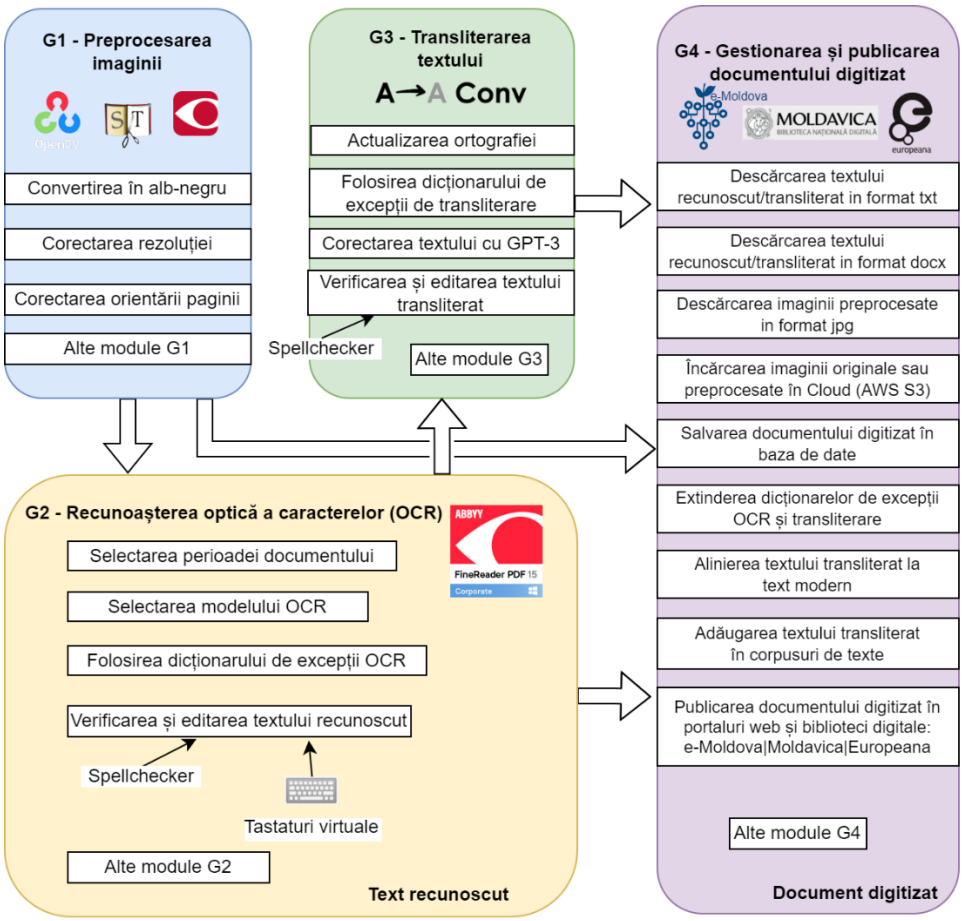



**Figura 3.1. Arhitectura platformei de digitizare.**

În continuare vom descrie modulele fiecărui grup funcțional în parte. Unele dintre aceste module sunt în faza de dezvoltare.

### Module de preprocesare a imaginii

Primul grup funcțional *G1* se ocupă de preprocesarea imaginii.

În contextul recunoașterii optice a caracterelor, preprocesarea se referă de obicei la pașii întreprinși pentru a pregăti o imagine cu text pentru ca motorul OCR să o poată analiza. Motorul

OCR poate avea uneori dificultăți în interpretarea corectă a imaginilor care sunt încețoșate, distorsionate sau au un contrast scăzut. Preprocesarea poate ajuta la îmbunătățirea preciziei OCR prin pregătirea imaginii pentru a fi mai potrivită pentru recunoaștere. Unele etape comune de preprocesare pentru OCR includ:

- Mărirea calității imaginii: Aceasta poate implica tehnici precum ajustarea contrastului sau a luminozității imaginii pentru a îmbunătăți citirea, sau șlefuirea imaginii pentru a reduce încețoșarea.
- Binarizarea: Aceasta implică convertirea imaginii în varianta alb si negru, ceea ce poate ajuta la îmbunătățirea contrastului și la „ușurarea" recunoașterii textului.
- Îndepărtarea zgomotului: Aceasta poate implica îndepărtarea pixelilor negri suplimentari din imagine care ar putea crea impedimente în funcționarea motorului OCR.
- Corectarea distorsiunii: Dacă imaginea nu este perfect aliniată, software-ul OCR poate avea dificultăți în interpretarea corectă a textului. Corectarea distorsiunii implică rotirea imaginii pentru a o alinia corect.

Prin preprocesarea imaginii înainte de a o trimite motorului OCR, de regulă, se poate îmbunătăți precizia si fiabilitatea procesului OCR.

În acest grup funcțional sunt integrate module de preprocesare din soft-urile Scan Tailor, FineReader 15, și pachetul Python – OpenCV. Scopul unității *G1* este de a pregăti documentul pentru OCR.

Un modul important din *G1* este binarizarea imaginii – aceasta fiind considerată o problemă de etichetare a pixelilor. Este definită ca o funcție care intensităților pixelilor imaginii de la intrare le pune în corespondență valorile *0 (negru)* sau *1 (alb)* în imaginea binarizată de la ieșire. Sunt două clase principale a metodelor de binarizare. Prima determină un prag global în imagine, pur și simplu un nivel de gri, și apoi atribuie valoarea 0 tuturor pixelilor cu valoarea mai mică decât acest prag și valoarea 1 celorlalți pixeli. Cea mai utilizată metodă din această clasă este metoda Otsu[60]. Cea dea doua clasă o constituie setul de metode cu prag local și determină un prag de binarizare diferit pentru fiecare pixel, în loc de a folosi un prag global pentru toată imaginea. Metodele cu prag local pot utiliza diverse tehnici pentru a determina pragurile de binarizare pentru fiecare pixel. Unele pot lua în considerare informații despre vecinii apropiați ai pixelului pentru a determina pragul, în timp ce altele pot utiliza statistici globale pentru a găsi un prag care se ajustează la nivelul local. După ce se determină pragurile de binarizare pentru fiecare pixel, acestea

[60] https://en.wikipedia.org/wiki/Otsu's_method

sunt folosite pentru a determina dacă fiecare pixel trebuie să fie tratat ca fiind alb sau negru în imaginea binară finală. O metodă din această clasă a fost introdusă de Sauvola și Pietikäinen[61]. Scopul acestor metode este de a face față problemelor care pot apărea atunci când se utilizează un prag global, cum ar fi iluminarea inegală a documentelor.

În Scan Tailor binarizarea este implementată prin următorii pași: egalizarea iluminării bazată pe metoda prezentată în lucrarea [117], netezirea Savitzky-Golay, binarizarea bazată pe metoda lui Otsu. Pasul final al metodei date de binarizare are ca scop și eliminarea marginilor întrerupte. Eliminarea marginilor întrerupte se referă la utilizarea unei imagini șablon drept referință, pentru a localiza și îndepărta marginile întrerupte din imaginea de intrare. În acest caz, imaginea șablon ar reprezenta o margine liniară fără întreruperi, iar algoritmul ar căuta aceasta în imaginea de intrare și ar înlocui orice margine întreruptă găsită cu o margine similară celei din șablon. Modulul de binarizare cu Scan Tailor este accesibil doar în varianta desktop a platformei de digitizare. Acest modul de binarizare a fost folosit la preprocesarea documentelor din secolele XVII și XVIII pentru crearea setului de date de antrenare și însăși antrenarea modelelor OCR.

Modulul de binarizare cu OpenCV a fost implementat în Python utilizând Filtrarea Bilaterală[62] descrisă în lucrarea [136], metoda lui Otsu și metoda de Potrivire a Șablonului[63] pentru eliminarea marginilor întrerupte. OpenCV vine cu funcția *cv.matchTemplate()*[64] pentru acest scop. Îndepărtarea marginilor întrerupte în modulul de binarizare cu OpenCV durează în jur de 10 secunde, de aceea în varianta curentă a acestui modul metoda de potrivire a șablonului este dezactivată.

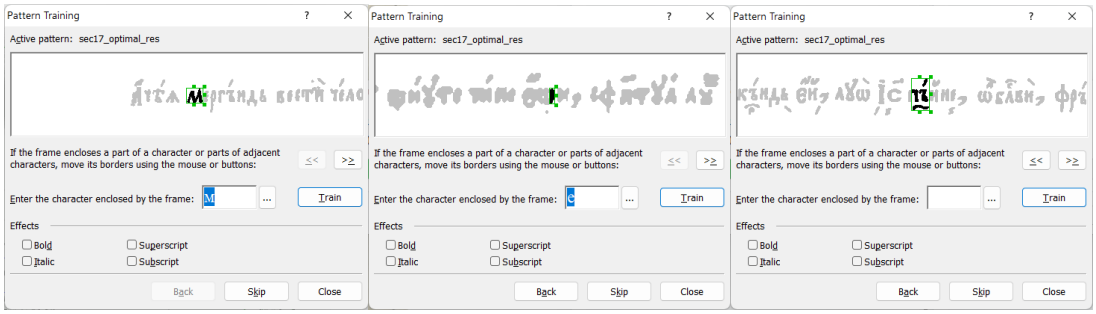
Modulul de binarizare cu FineReader se bazează pe clasa metodelor de prag local si se numește *binarizare adaptivă*[65]. Binarizarea adaptivă este o tehnică inovatoare utilizată în algoritmii de preprocesare a imaginilor de către ABBYY[66]. Aceasta a fost optimizată pentru a îmbunătăți calitatea imaginilor sursă anume pentru motorul OCR din FineReader 15. Această abordare de binarizare permite păstrarea a cât mai mulți pixeli de text pe imagini cu calitate scăzută și eliminarea zgomotului provocat de textul care străbate din versoul paginii.

Unul dintre modulele importante la etapa de preprocesare este cel care corectează rezoluția imaginilor. Rezoluția reprezintă numărul de pixeli prezenți într-o imagine și este importantă pentru ca modelele OCR să funcționeze eficient. Rezoluția poate varia în funcție de perioada în care a

[61] https://www.sciencedirect.com/science/article/abs/pii/S0031320399000552
[62] https://docs.opencv.org/4.x/d4/d86/group__imgproc__filter.html#ga9d7064d478c95d60003cf839430737ed
[63] https://docs.opencv.org/4.x/d4/dc6/tutorial_py_template_matching.html
[64] https://docs.opencv.org/4.x/df/dfb/group__imgproc__object.html#ga586ebfb0a7fb604b35a23d85391329be
[65] https://support.abbyy.com/hc/en-us/articles/360016460720-Adaptive-Binarization-and-Background-Filtering
[66] https://support.abbyy.com/hc/en-us/articles/360016570880-Binarization-Enhancements-in-ABBYY-Technologies

fost tipărit documentul, starea de uzură a acestuia și calitatea imaginii scanate. Pentru a obține rezultate optime, este important să se corecteze rezoluția imaginilor înainte de a le folosi pentru recunoașterea optică a caracterelor.

Pentru a obține rezultate optime în timpul procesării documentelor vechi, este important să se țină cont de rezoluția necesară pentru fiecare perioadă în parte. De exemplu, documentele din secolul XVII au nevoie de o rezoluție mai mare de 900 dpi (puncte pe inch) pentru a putea fi detectate ligaturile, dar mai mică de 1300 dpi pentru a nu se detecta mai multe caractere laolaltă (vezi figura 3.2c). Documentele din secolul XVIII pot avea nevoie de o rezoluție de la 300 dpi până la 600 dpi. Rezoluția suficientă pentru documentele din secolele XIX și XX este de 300 dpi, numită și "standard de aur al rezoluției"[67]. În cazul în care se „livrează" la recunoaștere o imagine cu o rezoluție diferită de cea a imaginilor folosite la antrenarea modelului OCR, pot apărea probleme sau anomalii în procesul de recunoaștere, mai ales pentru documentele vechi din secolul XVII, din cauză că în loc să fie detectat și segmentat caracterul întreg, se segmentează fie doar o porțiune de caracter sau mai multe caractere și rânduri laolaltă (vezi figura 3.2). Este important să se ia în considerare această diferență de rezoluție pentru a obține rezultate optime în timpul procesării documentelor.




a) b) c)

**Figura 3.2. Detectarea și segmentarea caracterelor în imagine cu: a) rezoluție optimală; b) rezoluție prea mică (96 dpi); c) rezoluție prea mare (2000 dpi); preluată dintr-un documente tipărit în secolul XVII cu caractere din alfabetul chirilic românesc. Detectarea caracterului nu înseamnă recunoașterea caracterului. Recunoașterea propriu-zisă a caracterului este etapa ce urmează după detectare și segmentare.**

Modulele de setare a rezoluției cu Scan Tailor și OpenCV oferă posibilitatea de setare manuală a rezoluției de către utilizator. În mod implicit Scan Tailor setează rezoluția imaginii de 600 dpi la ieșire. Am scris „rezoluția imaginii la ieșire", deoarece există și parametrul rezoluției la

[67] https://iscanner.com/scan-300-dpi-with-your-mobile-phone/

intrare a imaginii, ceea ce înseamnă că înainte ca Scan Tailor să înceapă alte procesări de imagine, imaginea încărcată deja trebuie să aibă o rezoluție potrivită. De exemplu, aplicarea opțiunii de ștergere a zgomotului ISO într-o imagine cu o rezoluție de 50 dpi s-ar putea să nu aibă nici un efect în Scan Tailor, de aceea uneori se cere și setarea rezoluției de intrare. Parametrul „rezoluția de intrare" sau pur și simplu rezoluție[68] în Scan Tailor este, în mod implicit, 600 dpi.

În FineReader 15, modulul de setare a rezoluției are și opțiunea de detectare optimală a rezoluției care oferă rezultate destul de bune pentru documentele din secolul XVII. Anume această opțiune a fost inclusă în *G1*.

Modulele de preprocesare din grupul *G1* sunt prezentate în tabelul 3.1. În mod implicit au fost integrate 3 motoare de preprocesare pentru a putea cuprinde cât mai multe module de preprocesare necesare.

**Tabelul 3.1. Module de preprocesare a imaginii și soft-urile integrate pentru implementarea modulelor din grupul funcțional *G1*.**

| Module de preprocesare a imaginii din *G1* | Motoare/instrumente de procesare integrate pentru implementare modulului |
|---|---|
| Selectarea motorului de preprocesare | Scan Tailor, FineReader 15, OpenCV |
| Binarizarea imaginii | Scan Tailor, FineReader 15, OpenCV |
| Setarea manuală a rezoluției imaginii | Scan Tailor, OpenCV |
| Corectarea automată a orientării paginii | FineReader 15, OpenCV, |
| Ștergerea zgomotului ISO | FineReader 15, OpenCV |
| Divizarea imaginii în mai multe pagini | FineReader 15 |
| Corectarea automată a rezoluției imaginii | FineReader 15 |
| Îndreptarea rândurilor de text | FineReader 15 |
| Corectarea manuală a orientării paginii | Scan Tailor |
| Curățarea petelor din imagine | Scan Tailor |
| Corectarea iluminării din imagine | Scan Tailor |
| Gestionarea grosimii caracterelor | Scan Tailor |

[68] https://helpmanual.io/help/Scan Tailor-cli/

### 3.2. Module de recunoaștere optică a caracterelor

Modulele din acest grup funcțional sunt utilizate pentru gestionarea modelelor OCR, recunoașterea propriu-zisă a documentului cât și editarea textului recunoscut.

Acțiunile din grupul G2 încep cu selectarea perioadei documentului. Deci, în mod obișnuit, utilizatorul cunoaște perioada istorică de tipărire a documentului care urmează să fie supus digitizării, mai mult ca atât, poate indica chiar și anul când a fost tipărit. De pe coperta cărții, dacă-i o carte; din sursa de unde a fost preluat documentul, prin verificarea istoricului documentului – dacă documentul are o istorie cunoscută, utilizatorul poate încerca să afle perioada în care a fost creat prin documentarea sursei; prin analiza stilului și a tipului de scriere, de exemplu, dacă documentul conține litere chirilice și latine în același cuvânt, poate fi din perioada anilor 1830-1860 când se tipărea cu alfabete de tranziție; prin verificarea datelor menționate în document, dacă documentul conține astfel de referințe; prin consultarea unui expert în domeniu (pentru aceasta poate fi adăugat un modul special în G2 sau chiar în G1 cu titlul „Află perioada documentului tău de la un expert", prin care se face legătura cu un specialist etc.) Totuși, nu este exclus și cazul, când utilizatorul are câteva imagini dintr-un document aleatoriu despre care nu știe nimic mai mult decât faptul că acest document este în grafie chirilică. Pentru astfel de cazuri ar fi util un modul de detectare automată a perioadei, care urmează să fie dezvoltat în cadrul platformei. O abordare ce poate fi utilă în soluționarea acestei probleme este experiența de detectare a fonturilor din documentele chirilice tipărite în secolul XVII, unde anumite modele de rețele neurale au fost antrenate să recunoască automat fontul documentului (vezi capitolul II, secțiunea 2.7). Un astfel de modul, care se ocupă de detectarea fonturilor din secolul XVII, este inclus în G2.

Cel mai important modul din G2 este selectarea modelului OCR. Utilizatorul are opțiunea de a alege un model OCR din cele adăugate implicit sau de a adăuga un model nou conform unor condiții stabilite în platformă. În mod implicit, sunt incluse în total 8 modele OCR. Un model pentru secolul XX, 2 modele pentru secolul XIX, 3 modele pentru XVIII și 2 modele pentru secolul XVII. Aceste modele OCR au fost obținute prin antrenarea motoarelor OCR din FineReader 12 și FineReader 15.

Modelul pentru secolul XX extinde prin *transfer learning*[69] modelul OCR pentru limba rusă integrat implicit în pachetul de limbi din FineReader 15. Acest model a fost antrenat preponderent să recunoască litera **ѫ**, care nu există în alfabetul rusesc. Setul de date conține 1040 de exemple de antrenare și se bazează pe următoarele surse: ziarul „Literatura și Arta" din anii 1988-1989, revista „Femeia Moldovei" din anii 1960-1970 și cartea „Folclor din părțile Codrilor"

[69] https://en.wikipedia.org/wiki/Transfer_learning

din 1973 [137]. Un avantaj mare al acestui model îl constituie un dicționar cu peste 447 de mii de cuvinte. Aceste cuvinte includ atât rădăcinile cuvintelor cât și formele lor flexionate. Generarea dicționarului de cuvinte pentru acest model se bazează pe un algoritm de backtracking care generează lexicul în grafie chirilică [138-139], utilizând anumite reguli de transliterare inversă, în cadrul căreia cuvintele românești moderne, scrise în grafie latină, au fost transpuse în echivalentele lor în grafie chirilică [139]. Acuratețea la nivel de caractere pentru acest model depășește 98%.

Luând în considerare că în secolul XIX documentele au fost tipărite cu diferite alfabete, au fost antrenate 2 modele OCR, unul bazat pe alfabetul chirilic românesc și altul bazat pe alfabetul de tranziție. Primul model antrenat cu caractere din alfabetul chirilic românesc se bazează pe seturi de date din cărțile „Legiuire" din anul 1818 și „Epistolariu românesc" din 1841. Setul de date de antrenare conține 2800 de exemple, iar dicționarul de cuvinte are peste 3000 de cuvinte. Al doilea model a fost antrenat pe seturi de date din documente tipărite cu alfabetul chirilic de tranziție. În particular, în acest scop a fost utilizată cartea lui Gheorghe Asachi „*Elemente de aritmetică"* din 1836.

Modelele pentru secolul XVIII au fost antrenate cu seturi de date preluate din documentele: *Fiziognomie,* din anul 1785; *Așezământ,* din 1786; și *De Obște Geografie,* din 1795. Dicționarul de cuvinte este comun pentru toate cele trei modele și conține peste 3000 de cuvinte. În particular, modelul bazat pe cartea *Fiziognomie* a fost antrenat cu 3600 de exemple de învățare și are un dicționar cu 1800 de cuvinte.

Antrenarea și evaluarea modelelor OCR pentru alfabetul chirilic românesc, utilizat în tipăriturile din secolul XVII a fost descrisă în secțiunea 2.5 din capitolul II. Anume pentru acest caz a fost identificată și problema fonturilor, pentru care au fost propuse soluții de clasificare. Vom menționa aici doar că modelul de bază pentru secolul XVII a fost antrenat pe un set de date format din 3668 de exemple de învățare, extrase din *Noul Testament de la Bălgrad*[70] din 1648, fiind însoțit de un dicționar cu 4582 de cuvinte. Acuratețea (la nivel de caractere) acestui model este de circa 95%.

Procesul de recunoaștere a documentelor poate fi împărțit în mai multe părți care pot fi efectuate în paralel pentru a îmbunătăți eficiența (viteza de procesare a mai multor documente). Utilizarea ABBYY Hot Folder[71] (în continuare Hot Folder), un agent care permite aplicarea modelului OCR necesar asupra unui dosar cu imagini, care vor fi procesate automat de motorul OCR FineReader 15 atunci când apar noi imagini în folder, poate ajuta la paralelizarea procesului.

[70] https://ro.wikipedia.org/wiki/Noul_Testament_de_la_Bălgrad
[71] https://help.abbyy.com/en-us/finereader/15/user_guide/hotfolder/

Acest lucru poate fi realizat prin utilizarea mai multor instanțe pentru fiecare model OCR, împărțind astfel imaginile preprocesate în mai multe dosare, ceea ce poate îmbunătăți eficiența atunci când mai mulți utilizatori lucrează simultan pe platformă. Rularea modelelor încărcate pe platformă este gestionată prin agentul Hot Folder.

Recunoașterea unei singure pagini cu text în medie durează 30 de secunde, cu toate că uneori recunoașterea unei astfel de pagini ar putea lua până la două minute. Aceasta se datorează faptului că instanțele create în Hot Folder verifică la fiecare minut (aceasta este opțiunea minimă de timp în Hot Folder) dacă au apărut imagini prelucrate noi în dosarele cu imagini prelucrate. Respectiv, dacă documentul a fost prelucrat în secunda 55, atunci procesul OCR va începe peste 5 secunde, iar dacă documentul a fost prelucrat în secunda 5 atunci, acesta va trebui să aștepte 55 de secunde până va porni procesul OCR. Un document PDF cu 50 pagini text se va recunoaște în circa 90 de secunde; un PDF cu 100 pagini text - 150 de secunde; PDF cu 360 pagini text - peste 385 de secunde (mai mult de 6 minute). Documentele-text în format PDF atestă durata de aproximativ 1.2 secunde per pagină. La PDF-urile cu imagini nu a fost observată o durată stabilă. Criteriul acurateței OCR la nivel de caractere și la nivel de cuvinte este analizat în capitolul 2 al tezei. De exemplu, modelul OCR pentru secolul XX ne dă o acuratețe la nivel de caractere de peste 98%; modele din secolul XVIII oferă peste 92% la nivel de cuvinte; iar modelul pentru secolul XVII oferă o acuratețe de peste 95% la nivel de caractere și dicționare de cuvinte, luând în considerare preprocesarea potrivită a imaginii, calitatea de scanare a documentului, uzura acestuia etc.

Pe lângă dicționarele de cuvinte folosite în interiorul motorului OCR (vezi secțiunea 2.3 din cap. II), pe platformă mai există și dicționare de excepții OCR care constau din tupluri formate dintr-o expresie care conține „ambiguități” de recunoaștere și varianta corectă a acestei expresii. Am utilizat sintagma “ambiguități de recunoaștere” din simplu motiv că unele litere au o similaritate grafică extrem de apropiată, iar uneori motorul OCR recunoaște cu o probabilitate foarte mare varianta greșită. În așa caz, dicționarul intern nu poate propune candidatul corect chiar dacă varianta corectă s-ar fi aflat în dicționar. De exemplu, litera **и** este confundată cu litera **н** în expresia „сърачїн”, respectiv dicționarul de excepții OCR ar putea conține excepția: (сърачї~~н~~, сърачїи). Pentru a trata astfel de situații am inclus în G2 o componentă de postprocesare OCR prin folosirea dicționarului de excepții. Dicționare de excepții sunt folosite și la transliterare, iar un modul similar avem și în G3.

În cadrul grupului funcțional G2 am inclus un modul de editare a textului recunoscut. Acest editor de text dispune de o tastatură virtuală web care își adaptează compoziția caracterelor în

funcție de perioada documentului, bazată pe componenta Javascript *simple-keyboard*[72]. Există, de asemenea, un modul dedicat gestionării tastaturilor virtuale pentru desktop. Implicit avem încărcată o tastatură virtuală pentru Windows cu caractere chirilice românești. Verificatorul ortografic din editorul de text este bazat pe componenta Javascript *simple-spellcheker*[73] și dicționarele de cuvinte utilizate în modulele OCR. Pe lângă verificatorul ortografic integrat în grupul funcțional G2, unele browsere, cum ar fi Mozilla Firefox[74] și Google Chrome[75], oferă servicii proprii de verificare ortografică, dar acestea nu sunt încă utile pentru textele în limba română scrise în grafia chirilică, deoarece nu permit adăugarea de dicționare personalizate. Cel puțin aceste servicii sunt utile pentru textul transliterat.

În următoarea secțiune vom vorbi despre modulele de transliterare incluse în grupul funcțional G3.

### 3.3. Module de transliterare a textelor

Grupul *G3* include module destinate transliterării și editării textului transliterat. Transliterarea este posibilă prin două căi. Prima cale este folosirea aplicației web de transliterare *AAConv*[76], iar cea de a doua posibilitate este folosirea aceleiași aplicații doar că în variantă desktop. O diferență notabilă între aceste două variante este că varianta web poate accepta doar până la 1.2MB de text la o singură procesare.

Perioada istorică a documentului păstrează starea din *G2* atunci când utilizatorul folosește modulul de selectare a acesteia, în caz contrar selectarea perioadei este accesibilă și din G3. Perioada documentului este de fapt un atribut al tuturor modulelor din G2 și G3.

Un modul important pentru utilizator îl constituie actualizarea ortografiei, unde la solicitare se iau în considerare normele scrierii limbii române moderne. Un exemplu este scrierea cu **â** (din **a)**. În procesul de transliterare, trecerea la scrierea cu "â" este realizată prin transliterare, doar dacă activăm opțiunea de actualizare a ortografiei, în caz contrar se păstrează scrierea originală. Vom aminti, că potrivit recomandărilor Academiei Române, litera "î" va fi întotdeauna scrisă la începutul și sfârșitul cuvântului ("început", "înger", "în", "întoarce", "a coborî", "a urî"). În interiorul cuvântului, de obicei este scris "â" ("cuvânt", "a mârâi"). Totuși, există câteva excepții pentru această regulă. Cuvintele formate prin prefixarea cuvintelor care încep cu litera "î" vor

[72] https://hodgef.com/simple-keyboard/
[73] https://www.npmjs.com/package/simple-spellchecker
[74] https://www.mozilla.org/ro/firefox/
[75] https://www.google.com/chrome/
[76] https://translitera.cc/

păstra acest "î" în interior. De exemplu, "neîmpăcat", "neîngrijit", "preîntâmpinat", "reînarmat". Aceeași regulă se va aplica și cuvintelor compuse: "bineînțeles", "semiînchis" etc. [109]

La transliterare ne mai întâlnim cu abateri de la normele generale care nu pot fi controlate prin reguli prestabilite, iar pentru aceasta grupul G3 include un modul pentru folosirea dicționarului de excepții. Dicționarul cu excepții de transliterare păstrează cuvinte care nu pot fi transliterate corect utilizând doar regulile de transliterare. De exemplu, cuvântul "амязэ" conform regulilor de transliterare trece in "amează", varianta corectă fiind "amiază", aceasta regăsindu-se în dicționarul respectiv. În acest modul este posibilă gestionarea listei de excepții. Excepțiile se tratează după transliterarea textului din grafia chirilică în cea latină conform regulilor, dar înainte de vizualizarea și verificarea textului în editorul de texte. Mai multe excepții provin de la scrierea diferită a cuvintelor de origine străină, în special a substantivelor proprii.

De asemenea, grupul *G3* împărtășește același modul de editare a textului cu grupul G2, iar tastatura virtuală și dicționarele de cuvinte pentru verificatorul ortografic sunt adaptate la textul transliterat. Aici se are în vedere că tastatura virtuală conține literele alfabetului românesc modern, iar dicționarul de cuvinte este scris cu alfabetul românesc modern.

Un modul experimental este corectarea textului transliterat cu un sistem de inteligență artificială de top. Acest sistem se numește GPT-3[77] dezvoltat de OpenAI[78]. Modelele de învățare automată, numite și modele lingvistice din GPT-3 pot rezolva probleme de prelucrare a limbajului natural, precum elaborarea rezumatelor, parafrazarea textului, traducerea automată, clasificarea textului, transformarea textului din limbaj natural în codul unui limbaj de programare, corectarea textului etc. GPT-3 rezolvă aceste probleme cu o acuratețe foarte bună, atât pentru limba engleză cât și pentru limba română. În modulul implementat în G3 folosim modelul *text-davinci-003* (în continuare *davinci*) pentru corectarea textului recunoscut prin furnizarea condiției: *corectează textul X,* unde X este textul transliterat. Modelul poate procesa până la 4000 de „tokenuri" per cerere. Un token[79] in *davinci* are in medie 4 caractere din engleza, iar 100 de tokenuri ar fi echivalentul a aproximativ 75 de cuvinte. Un moment crucial la corectare este faptul că acest model nu păstrează varianta originală a expresiilor arhaice din textul transliterat. Un exemplu corectat de *davinci* este „*Cunoscând și Înțelegând măria Ta, noi, românii care suntem în țara măriei Tale, nu avem nici Testamentul cel Nou, nici cel Vechi în limba noastră.*" pentru expresia transliterată „*cum să cunoaște că văzând și Înțelegând măria ta, că noi rumânii carii sântem în țara măriei tale. nu ave m neci Testamentul cel nou, neci cel vechiu de plin întru limba noastră,*".

[77] https://en.wikipedia.org/wiki/GPT-3
[78] https://en.wikipedia.org/wiki/OpenAI
[79] https://help.openai.com/en/articles/4936856-what-are-tokens-and-how-to-count-them

Luând în considerare că acest text a fost scris în secolul XVII, practic ceea ce a făcut modelul GPT-3 seamănă mai mult cu o aliniere a textului vechi la textul modern. Însă acesta nu este cazul și pentru textele din secolul XX unde textul diferă nesemnificativ față de textele moderne. În acest caz, *davinci* corectează suficient de bine. După cum am mai spus, corectarea textului cu GPT-3 este un modul experimental care trebuie exploatat mai mult timp pentru a putea face concluzii mai generale.

Alte module din G3 se referă la elemente de corectare „stilistică" a textului transliterat. Aici s-ar putea include înlocuirea apostrofului cu cratimă (exemplu: *s'ar* cu *s-ar*) sau ștergerea cratimei dintr-un cuvânt care se află la sfârșit de rând (exemplu: *ră-\npirea* trece în *răpirea*).

În continuare vom vorbi despre modulele din grupul funcțional G4.

## 3.4. Module de gestionare a documentului digitizat

Documentul digitizat se referă la imaginile originale și preprocesate, textele recunoscute și cele transliterate. Grupul funcțional G4 include module de gestionare a documentului digitizat.

Un modul din G4 îl constituie descărcarea imaginilor preprocesate. Utilizatorul poate descărca imaginile pe dispozitivul personal pentru necesități ulterioare. Formatul imaginilor descărcate este JPG.

Similare sunt și modulele de descărcare a textelor recunoscute și transliterate. Formatele fișierelor descărcate includ TXT și DOCX. Varianta DOCX nu este altceva decât o împachetare a textului crud (fără formatare), fără a păstra stilurile sau ilustrațiile din documentul original.

Un modul important din G4 este salvarea documentului digitizat în baza de date a platformei. Pe lângă stocarea textelor și a link-urilor către fișiere, se mai stochează și obiectul digitizat care reprezintă un obiect[80] JavaScript cu ajutorul căruia putem păstra starea fiecărui pas făcut prin aplicația de digitizare descrisă în următoarea secțiune. Obiectul include parametrii de preprocesare, parametrii de recunoaștere și transliterare, textul recunoscut și editat, textul transliterat și editat, etc.

Pentru a face față mulțimii de imagini încărcate pe platformă, am inclus în G4 un modul de încărcare a documentelor originale și a imaginilor preprocesate în *Cloud*. La dezvoltarea acestui modul am creat un *bucket* utilizând serviciul Amazon S3[81] (Simple Storage Service) – un serviciu de stocare în Cloud dezvoltat de Amazon Web Services (AWS). Un bucket Amazon S3 este un container pentru stocarea fișierelor în Amazon S3. Putem stoca orice tip de fișier într-un *bucket*,

[80] https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Object
[81] https://aws.amazon.com/s3/

de la imagini, muzică și video la aplicații, site-uri web etc. Un bucket este identificat prin nume unic în Amazon S3 și poate fi accesat prin intermediul unei adrese URL care începe cu "https://s3.amazonaws.com/". Acesta poate fi configurat cu diferite opțiuni de securitate pentru a proteja conținutul lor și poate fi utilizat pentru a distribui conținut prin intermediul Amazon CloudFront[82], o rețea de distribuție de conținut (CDN[83]).

Un set de module foarte importante din G4 se referă la publicarea documentului digitizat. Conștientizând existența problemei dreptului de autor și optând pentru soluționarea ei cu respectarea actelor normative, în cele ce urmează vom propune doar soluții tehnice care ar putea funcționa în prezumția soluționării aspectelor juridice. Modulul de publicare este axat pe portalul eMoldova[84] [140], în particular bazat pe un portlet[85] numit Tezaurul Național Digital[86] (în continuare *digi*). Acest portlet conține resurse din tezaurul digital moldovenesc, dar și unele servicii simplificate de digitizare și gestionare a documentelor digitizate (numite *articole digitale* în digi). Digi include grupuri de lucru speciale pentru a oferi publicului larg articole digitale cât mai calitative din punct de vedere al corectitudinii textului recunoscut/transliterat. Grupurile de lucru se împart în două. Există grupul de lucru „digizori", cei care încarcă un document digitizat pe platformă, și grupul de lucru „redactori" cei care verifică și aprobă articolul digital spre publicarea finală. Pentru etichetarea cu metadate, *digi* oferă o selecție mare de etichete, ce provin din metadatele diferitor publicații. Aceasta permite căutarea și filtrarea simplă a articolelor în baza tipului documentului (reviste, cărți, manuscrise), un an ori o perioadă concretă. La etichetare se includ de asemenea etichete interne. Articolele marcate cu aceste etichete au acces limitat, fiind vizibile doar de utilizatorii cu roluri sau privilegii speciale. Un exemplu este eticheta „Spre redactare" care se folosește pentru articolele încărcate de digizor. Articolele etichetate cu eticheta respectivă sunt accesibile doar pentru grupul de redactori. Atunci când un redactor aprobă articolul digital spre publicare către toți utilizatorii, se va înlocui eticheta „Spre redactare" cu „Verificat de redactor". Se prevede adăugarea în *digi* a condiției ca mai mulți redactori să aprobe un document digital spre publicarea finală.

Alte caracteristici ale digi sunt: notificarea grupurilor de lucru; crearea schițelor de articole; versionarea redactărilor efectuate asupra unui articol digital; adăugarea semnelor de carte pentru monitorizarea eficientă a articolelor focalizate de utilizator; un sistem pentru premierea utilizatorilor pentru activitate; aprecierea articolelor digitale; adăugarea de comentarii privind un

[82] https://aws.amazon.com/cloudfront/
[83] https://en.wikipedia.org/wiki/Content_delivery_network
[84] https://emoldova.org/
[85] https://fr.wikipedia.org/wiki/Portlet
[86] https://digi.emoldova.org/

articol digital sau altul; îmbinarea mai multor articole digitale într-un singur articol digital (poate fi util când se digitizează o carte sau o revistă), etc. În figura 3.3 este prezentată o pagină a unui articol digital care include documentul original, textul recunoscut, textul transliterat și un meniu lateral în partea dreaptă a paginii pentru a comuta conținutul dorit.

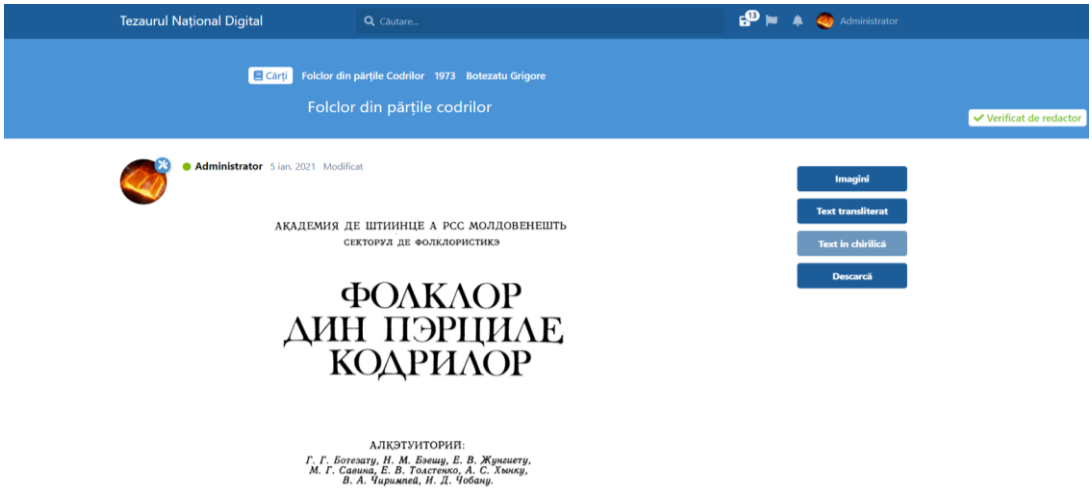



**Figura 3.3. Pagina unui articol digital publicat în digi:**
**https://digi.emoldova.org/d/17-folclor-din-partile-codrilor**

### 3.5. Aplicație de digitizare

Aplicația de digitizare este o instanță demonstrativă a platformei de digitizare. Scopul elaborării acestei aplicații este demonstrarea funcționalului unor module din platformă. Aplicația permite digitizarea unui document în 7 pași, unii dintre care fiind opționali, cu posibilitatea de a fi omiși. Procesul de traversare a acestor pași îl vom numi *ciclu de digitizare*.

În figura 3.4 este afișat un fragment din interfața grafică a platformei cu pașii din ciclu de digitizare – în stânga este afișată interfața grafică pentru ciclul de digitizare în faza incipientă a unui document care urmează a fi digitizat, în mijloc este afișată interfața grafică pentru un document care se află la pasul 4 din ciclul de digitizare – pasul cu verificarea și editarea textului recunoscut, iar în dreapta este afișată interfața grafică pentru un ciclu de digitizare complet, atunci când documentul a ajuns la ultimul pas (pasul 7) din ciclu de digitizare. Interfața grafică cu pașii din ciclu de digitizare își schimbă starea în funcție de completarea fiecărui pas în parte. Un pas parcurs este marcat de culoarea albastru-închis a fundalului și o bifă într-un cerculeț verde. Fundalul albastru-deschis al unui pas de digitizare indică starea curentă din ciclul de digitizare a documentului.


**Figura 3.4. Trei stări ale unui ciclu de digitizare a unui document.**

În continuare vom descrie fiecare pas din ciclul de digitizare.

### Încărcarea fișierelor

Într-un singur ciclu de digitizare pot fi prelucrate unul sau mai multe fișiere. Se acceptă următoarele tipuri de fișiere: **png, jpeg, tiff**. Mărimea totală a tuturor fișierelor încărcate nu trebuie să depășească 700MB, iar mărimea fiecărui fișier are limita de 100MB. Fișierele pot fi selectate din calculatorul utilizatorului. Atunci când vor fi selectate două sau mai multe fișiere, se va lua în considerare că toate aceste fișiere vor fi procesate cu aceleași opțiuni de procesare, prin urmare utilizatorul se va asigura că fișierele încărcate sunt din aceeași perioadă, au unul și același alfabet și necesită aceleași opțiuni de preprocesare a imaginii. Dacă utilizatorul are seturi de documente din mai multe perioade sau care necesită opțiuni diferite de preprocesare a imaginii, atunci aceste seturi vor fi digitizate în diferite cicluri de digitizare.

Unii pași de digitizare au secțiune „Info” sau „?”. Această secțiune oferă informații adiționale referitoare la pasul dat de digitizare.

Utilizatorul poate comuta pașii prin două căi. Prima cale include un buton adițional de trecere la următorul pas care apare după ce se îndeplinesc acțiunile pasului curent. Cea dea doua cale se realizează  prin comutarea pașilor utilizând butoanele meta de comutare.

**Preprocesarea imaginilor încărcate**

La acest pas utilizatorul poate sa aleagă motorul de preprocesare și opțiunile de preprocesare necesare paginilor sale. La momentul dat (20 decembrie 2022) sunt disponibile 3 motoare de preprocesare: Scan Tailor, FineReader 15 și OpenCV. Modulele de preprocesare au fost discutate în secțiunea 3.2 din acest capitol.

*Scan Tailor* este primul motor de preprocesare din lista de opțiuni din pasul 2 și oferă cea mai largă gamă de opțiuni din toate cele trei motoare propuse. Spre deosebire de celelalte motoare, se propun două modalități de utilizare a motorului Scan Tailor pentru preprocesarea imaginii (vezi figura 3.5): a) Desktop - cu integrarea propriu-zisă a aplicației Scan Tailor, împreună cu instrucțiunile pentru utilizator; b) Web - cu opțiuni de bază de preprocesare a imaginii recomandate prin folosirea API-ului *Scan Tailor-cli* din varianta web a platformei. Modalitatea desktop de preprocesare cu Scan Tailor oferă mai multe opțiuni de preprocesare, inclusiv opțiuni de tăiere manuală a imaginii, selectare manuală a conținutului, curățare manuală a petelor din imagine, etc. În varianta Web a platformei sunt selectate doar opțiunile cele mai necesare, fără de a se lua în considerare imaginile excepționale (care au nevoie de unele setări speciale).

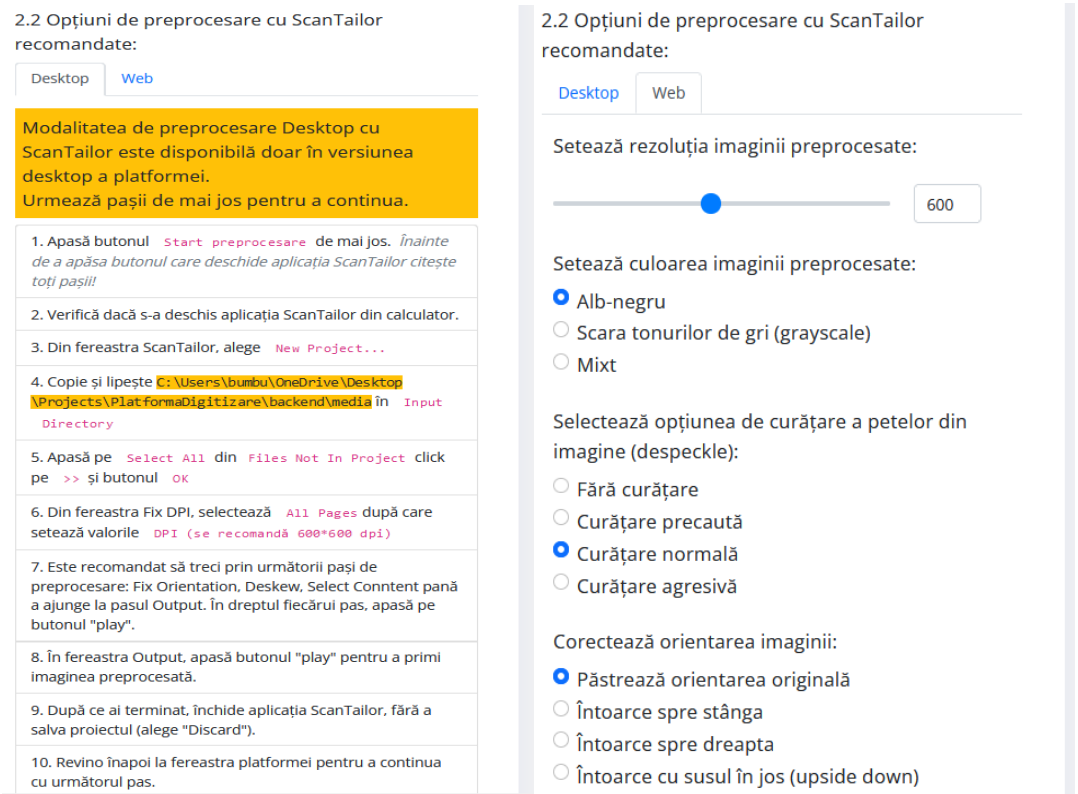



**Figura 3.5. Două modalități de preprocesare cu Scan Tailor: în stânga - preprocesarea Desktop, în dreapta - opțiuni de preprocesare Web.**

Din motoarele de procesare propuse în platforma de digitizare, Scan Tailor (desktop) ar fi varianta cea mai bună pentru preprocesarea de documente vechi din secolele XVII și XVIII, în special acolo unde este nevoie de experimentarea cu diferite valori de rezoluție și uneori de îngroșarea caracterelor.

Următorul motor de preprocesare integrat în aplicație este motorul de preprocesare din **FineReader 15.** Pot fi testate deocamdată funcțiile de bază de preprocesare a imaginii, cum ar fi: *corectarea rezoluției imaginii* sau *detectarea rezoluției optimale pentru imaginea dată*; *corectarea automată a orientării paginii*; *convertirea imaginii în alb-negru*; *reducerea zgomotului ISO din imagine*; și *îndreptarea rândurilor de text*.

Motorul **OpenCV** permite setarea manuală a rezoluției imaginii preprocesate, dar oferă și filtre de curățare a imaginii de pete și reducere a zgomotului ISO integrate într-o singură opțiune, ceea ce poate simplifica lucrul utilizatorului, dar totodată nu se asigură un control precum în cazul lui FineReader sau Scan Tailor. O imagine procesată cu OpenCV este afișată în figura 3.6.

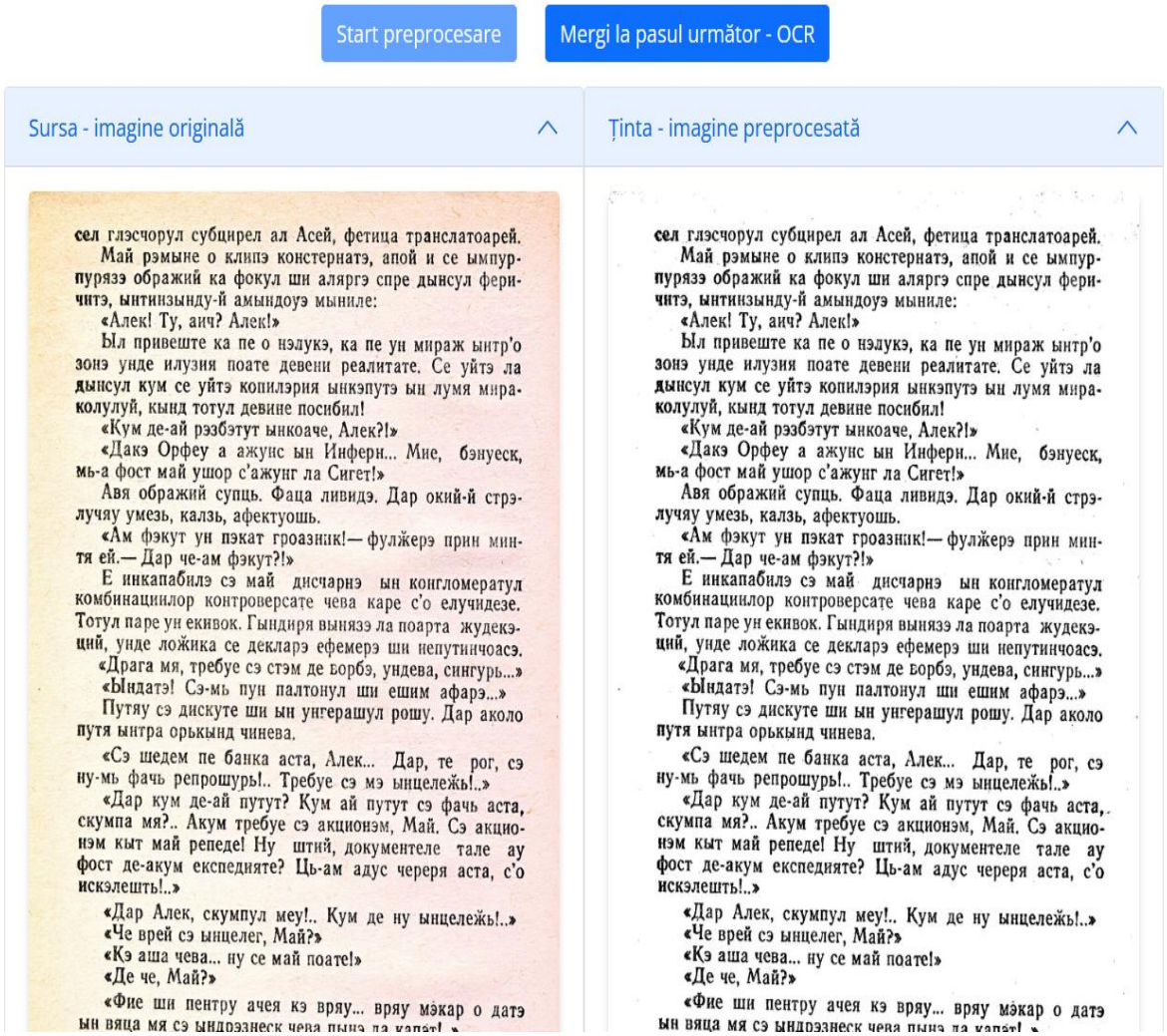



**Figura 3.6. O imaginea cu documentul original (stânga) și imaginea preprocesată cu motorul OpenCV în pasul 2.**

Următorul pas îl constituie recunoașterea optică a caracterelor.

### Recunoașterea optică a caracterelor

La acest pas utilizatorul selectează perioada documentului și modelul pe baza căruia se va recunoaște documentul (figura 3.7). Un exemplu de document recunoscut este afișat în figura 3.8.

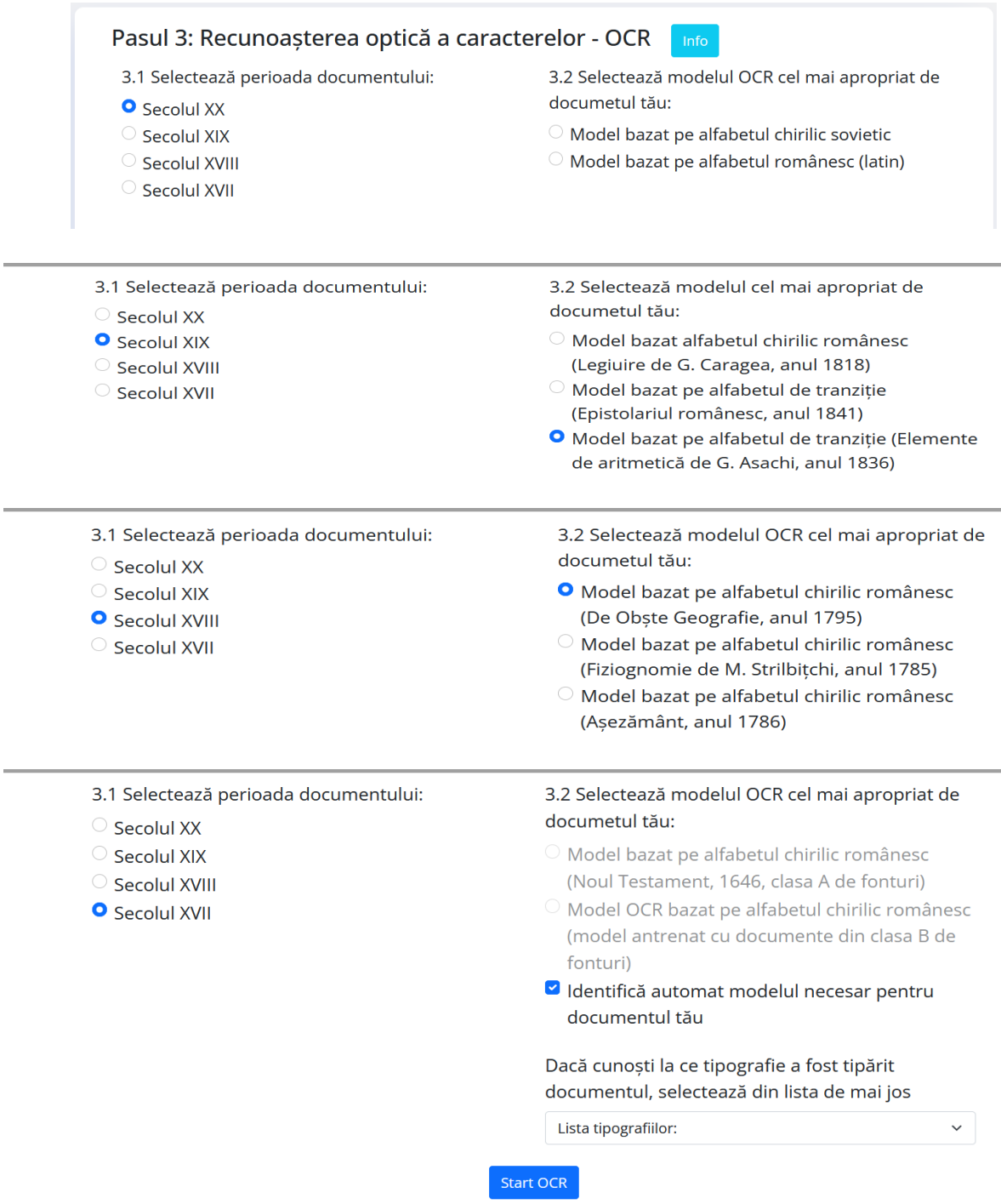



**Figura 3.7. Opțiuni disponibile în pasul 3.**

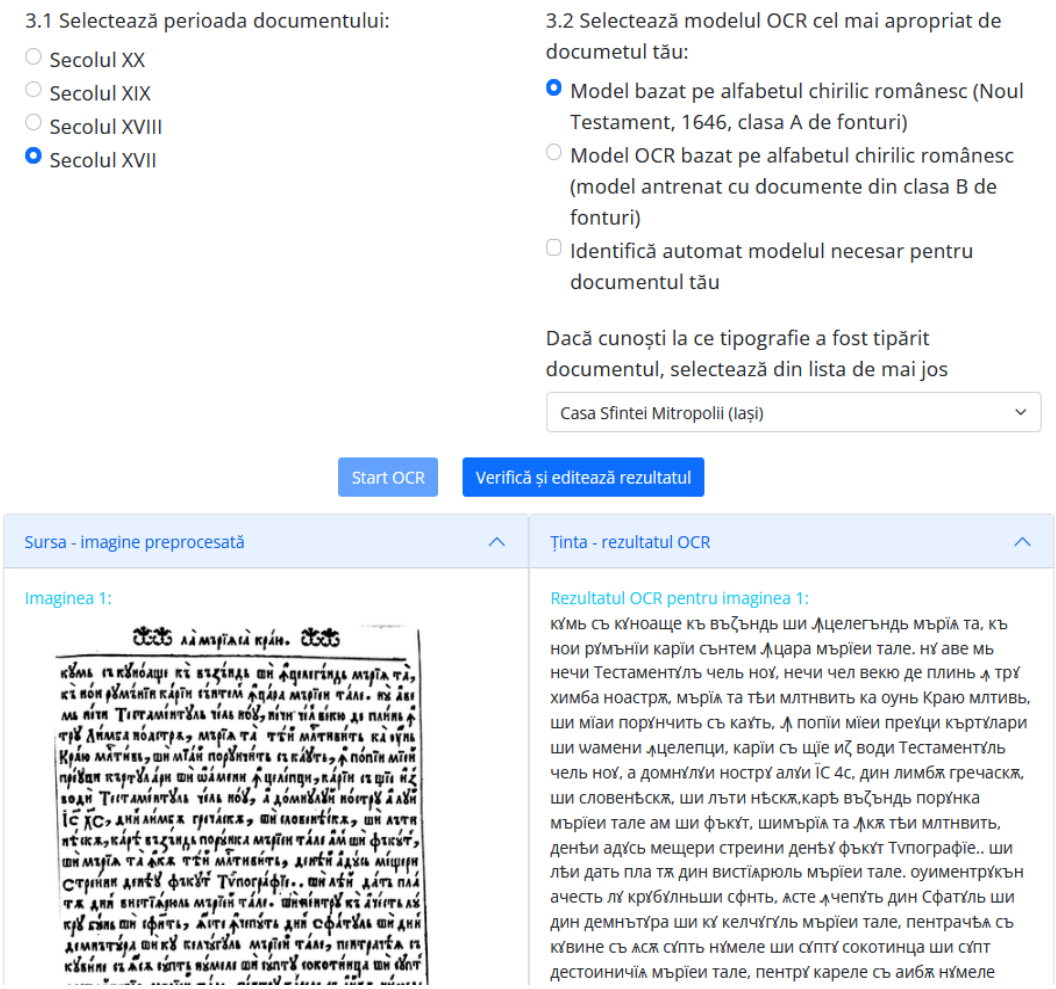



**Figura 3.8. Executarea pasului 3 din aplicația de digitizare.**

Următorul pas este verificarea și editarea textului recunoscut.

### Verificarea și editarea textului recunoscut

Acest modul permite prelucrarea manuală a textului obținut din pasul anterior. Sunt integrate modulele pentru editarea textuală, precum tastatură virtuală pentru fiecare alfabet în parte și dicționare de cuvinte pentru verificarea ortografiei. Un exemplu este afișat în figura 3.9.


**Figura 3.9. Verificare textului recunoscut.**

La următorul pas, pasul 5, se efectuează transliterarea textului recunoscut.

### Transliterarea textului recunoscut și editat

La acest pas sunt integrate modulele de transliterare, precum și cele de actualizare a ortografiei și folosirea dicționarului de excepții. Un exemplu este afișat în figura 3.10.


**Figura 3.10. Transliterarea unui text recunoscut.**

Următorul pas este editarea textului transliterat.

### Verificarea și editarea textului transliterat

Pasul de verificare a textului transliterat este similar cu cel de verificare a textului recunoscut operând cu aceleași dicționare de cuvinte precum și în cazul verificatorului ortografic.

Ultimul pas, al șaptelea din aplicația de digitizare, include module privind salvarea rezultatelor.

### Salvarea rezultatelor

În urma îndeplinirii pașilor 1-6 am obținut mai multe rezultate, texte și imagini, care împreună constituie un document digitizat. La acest pas sunt incluse module din grupul G4 pentru a descărca textul recunoscut, textul transliterat și imaginile preprocesate.


**Figura 3.11. Pasul 7 din aplicația de digitizare.**

Vom nota că în acest exemplu utilizatorul  a omis verificarea și corectarea textului.

## 3.6. Concluzii la capitolul 3

În acest capitol am definit și am descris arhitectura, modulele și aplicația de digitizare disponibile în cadrul platformei de digitizare. Platforma include module necesare pentru a realiza patru sarcini principale referitoare la digitizarea documentelor vechi românești: preprocesarea imaginii, recunoașterea documentelor, transliterarea textului recunoscut din grafie chirilică în latină, gestionarea și publicarea documentelor digitizate. Platforma poate fi folosită atât ca aplicație web, cât și în calitate de aplicație desktop [135]. Varianta desktop a platformei conține opțiuni avansate de preprocesare a imaginii, bazate pe utilizarea directă a instrumentului software *Scan Tailor*. Aplicația de digitizare integrată pe platformă permite digitizarea în 7 pași a documentului, iar durata de timp pentru un ciclu complet de digitizare variază între 2 și 15 minute, în funcție de volumul documentului. Timpul necesar pentru redactarea textului depinde de numărul de erori, mărimea documentului și viteza de redactare. Utilizatorul obține la final un *document digitizat*, care conține textul recunoscut, textul transliterat, imaginea preprocesată. Această platformă include module utile bibliotecilor, editurilor, cercetătorilor din diverse domenii, care dețin colecții de documente în limba română tipărite cu caractere chirilice. Important este faptul că platforma poate fi extinsă cu module de digitizare și pentru alte limbi.

# CONCLUZII GENERALE ȘI RECOMANDĂRI

Suportul procesului de revitalizare a patrimoniului cultural-istoric rămâne o problemă, actualitatea și importanța căreia constituie o prioritate menționată în mai multe documente de politici ale țărilor europene. Prin realizarea obiectivelor stabilite în cadrul prezentei teze au fost aduse anumite contribuții pentru facilitarea digitizării și transliterării textelor tipărite în limba română cu caractere chirilice, fiind acoperit un segment temporal al ultimelor patru secole. Prin analiza și dezvoltarea metodelor și instrumentelor folosite în preprocesarea imaginilor, elaborarea modelelor OCR etc., înglobate în platforma de digitizare este înlesnit accesul la documentele vechi chirilice românești, deschizând noi oportunități pentru cercetarea și valorificarea resurselor culturale și istorice.

Studiul rezultatelor obținute permite formularea următoarelor concluzii generale:

- Analiza instrumentelor și metodelor de digitizare a documentelor istorice relevă o multitudine de metode, instrumente, resurse și platforme disponibile pentru preprocesarea, recunoașterea, postprocesarea și transliterarea documentelor istorice, care se deosebesc prin acuratețea și eficiența operațională [28-30, 35-17, 40, 58, 49, 65, 80, 83, 93, 99].
- Metodele de recunoaștere bazate pe antrenarea modelelor OCR pe imagini cu linii de text sporesc viteza de antrenare a motoarelor OCR, contribuind astfel la accelerarea procesului de digitizare, acordându-i acestuia un caracter de masă [53-54].
- În rezultatul adaptării componentelor sistemului software FR15 pentru recunoașterea tipăriturilor vechi românești s-a constatat că acuratețea modelului crește semnificativ odată cu creșterea numărului de pagini de antrenare. Evaluarea procedurii de învățare în cadrul unui proces iterativ a demonstrat că odată cu creșterea numărului datelor de antrenare crește semnificativ și acuratețea modelului, atingând valori acceptabile (0.96 în cazul operării cu dicționar și 0.95 în cazul operării fără dicționar) la nivel de recunoaștere corectă a caracterelor chiar și după un număr nu prea mare de pagini (5-7 pagini). La nivel de cuvinte valoarea acurateței este mai mică, fapt ce denotă necesitatea utilizării unui număr mai mare de pagini pentru instruire.
- Pentru procesarea imaginilor din documentele vechi putem utiliza instrumente existente de preprocesare, suplimentându-le pe cele incorporate în FR15 cu posibilitățile oferite de Scan Tailor, în special pentru îngroșarea caracterelor, netezirea Savitzky-Golay și eliminarea marginilor întrerupte.

- Algoritmul de clasificare a fonturilor, dezvoltat prin crearea și antrenarea unei rețele neuronale multistrat [127], a demonstrat o acuratețe de peste 96%.
- Transliterarea din alfabetul chirilic român în cel modern cu utilizarea regulilor dependente de context se efectuează cu o acuratețe care întrece 98%.
- Instrumentarul de aliniere elaborat efectuează alinierea textelor vechi la cele moderne prin evaluarea și crearea similitudinii textelor paralele diacronice, bazându-se pe similaritatea șirurilor de caractere. Acest instrumentar facilitează crearea unui corpus paralel diacronic [129].
- Platforma de digitizare, arhitectura, modulele și aplicațiile căreia au fost elaborate în cadrul tezei, incluzând instrumente de preprocesare a imaginilor, modele OCR, aplicații pentru transliterarea din grafie chirilică în cea latină, module de editare a textelor recunoscute/transliterate, permite realizarea sarcinilor principale referitoare la digitizarea documentelor vechi românești într-un mod eficient și rapid [134-135].
- Platforma de digitizare poate fi folosită ca aplicație web sau desktop și poate fi extinsă pentru a include module de digitizare pentru alte limbi. Această platformă este utilă pentru biblioteci, edituri și cercetători care dețin colecții de documente în limba română tipărite cu caractere chirilice. De rând cu acestea, existența unei astfel de platforme, în special în versiunea web, facilitează accesul la tezaurul literar-istoric și pentru publicul larg.

Recomandările pentru viitoarele cercetări și dezvoltări în domeniul digitizării documentelor chirilice românești ar putea include:

- Îmbunătățirea continuă a modelelor OCR și a algoritmilor de transliterare, prin integrarea unor tehnici noi și avansate în domeniul prelucrării limbajului natural și învățării automate, pentru a crește precizia și eficiența procesului de recunoaștere și transliterare.
- Elaborarea unei interfețe simple de antrenare a modelelor OCR în masă pentru a deschide accesul pentru cât mai mulți utilizatori. Altfel, vom putea construi modele OCR practic pentru intervale de timp scurte și pentru majoritatea tipografiilor.
- Extinderea platformei de digitizare pentru a include și alte tipuri de documente, cum ar fi manuscrisele, hărțile sau ilustrațiile, pentru a permite accesul la o varietate mai mare de resurse culturale și istorice.

- Integrarea platformei de digitizare cu alte instrumente și resurse digitale, cum ar fi biblioteci digitale, arhive și baze de date, pentru a facilita colaborarea între cercetători și pentru a pune la dispoziție informații și resurse adiționale.

## BIBLIOGRAFIE


**[1]** Recomandarea Comisiei din 27 octombrie 2011 privind digitizarea și accesibilitatea online a materialului cultural și conservarea digitală (2011/711/UE) – În: Jurnalul Oficial al Uniunii Europene, 29.10.2011, https://eur-lex.europa.eu/legal-content/RO/TXT/PDF/?uri=CELEX:32011H0711&from=EN (Accesat 24.03.2023).

**[2]** Centru de Competență în Digitizare „IMPACT", http://www.digitisation.eu/community/map-of-the-digitisation-landscape/ (Accesat 5.08.2022).

**[3]** Proiectul „Gutenberg", http://www.gutenberg.org/ (Accesat 23.03.2023).

**[4]** Proiectul de Digitizare „Hathi Trust", https://www.hathitrust.org/. (Accesat 23.03.2023).

**[5]** Proiectul „Million Book Collection", http://ulib.isri.cmu.edu/. (Accesat 23.03.2023).

**[6]** Proiectul de Digitizare „Google Books", https://books.google.com/ (Accesat 23.03.2023).

**[7]** NIGGEMANN, E., DE DECKER, J., LÉVY, M. The new renaissance. In: *Raportul 'comité des sages'.* Grup de reflecție pentru aducerea online a patrimoniului cultural al Europei. Bruxelles, Comisia Europeană, 2011, 45 p.

**[8]** BALK, H., CONTEH, A. IMPACT: centre of competence in text digitisation. In: *Proceedings of the 2011 Workshop on Historical Document Imaging and Processing.* ACM, 2011, pp. 155–160.

**[9]** MANDELL, L. C., NEUDECKER, C., ANTONACOPOULOS, A., GRUMBACH, E., AUVIL, L., CHRISTY, M. J., HEIL, J. A., SAMUELSON, T. Navigating the storm: IMPACT, eMOP, and agile steering standards. In: *Digital Scholarship in the Humanities*, vol. 32, no. 1, pp. 189–194, 2017.

**[10]** BOIAN, E., CIUBOTARU, C., COJOCARU, S., COLESNICOV, A., MALAHOV, L. DIGITIZAREA, RECUNOAȘTEREA ȘI CONSERVAREA PATRIMONIULUI CULTURAL-ISTORIC. *Revista Akademos*, nr. 1 (32), martie 2014, pp. 61-68.

**[11]** MORUZ, M., IFTENE, A., MORUZ, A., CRISTEA, D. Semi-automatic alignment of old Romanian words using lexicons. In: *Proceedings of the 8th International Conference „Linguistic resources and tools for processing ofthe Romanian language"*, Iași, Editura Universității „A.I. Cuza", 2012, pp. 119-125.

**[12]** HAUG, D. T. T., JØHNDAL, M. L. Creating a Parallel Treebank of the Old Indo-European Bible Translations. In: *Caroline Sporleder and Kiril Ribarov (eds.). Proceedings of the Second Workshop on Language Technology for Cultural Heritage Data (LaTeCH 2008)*, 2008, pp. 27-34.

**[13]** VITAS, D., KRSTEV, C., OBRADOVIĆ, I., POPOVIĆ, L., PAVLOVIĆ-LAŽETIĆ, G. *In: International Workshop on Balkan Language Resources and ToolsAn Overview of Resources and Basic Tools for the Processing of Serbian Written Texts*, Thessaloniki, Greece, 2003, pp. 97-104.

**[14]** PAVLOV, R., BOGDANOVA, G., PANEVA-MARINOVA, D., TODOROV, T., RANGOCHEV, K. Digital archive and multimedia library for bulgarian traditional culture and folklore. In: *International Journal "Information Theories and Applications".* Vol. 18, Number 3, 2011, pp. 276-288.

**[15]** INDERMÜHLE, E., LIWICKI, M., BUNKE, H. Recognition of Handwritten Historical Documents: HMM-Adaptation vs. Writer Specific Training. In: *Proceedings of the 11th International Conference on Frontiers in Handwriting Recognition, August 19-21, 2008,* Concordia University Montreal, 2008, pp. 186-191.

**[16]** КОРНИЕНКО, С., АЙДАРОВ, Ю., ГАГАРИНА, Д., ЧЕРЕПАНОВ, Ф., ЯСНИЦКИЙ, Л. Программный комплекс для распознавания рукописных и старопечатных текстов. *«Информационные Ресурсы России»* №1, 2011, pp. 35-37.

**[17]** „Concept of Digital Heritage." In: *UNESCO*. https://en.unesco.org/themes/information-preservation/digital-heritage/concept-digital-heritage (Accesat: 1.04.2023).

**[18]** BRUMANN, C. Cultural Heritage. *International Encyclopedia of the Social & Behavioral Sciences (Second Edition), Elsevier, 2015*, pp. 414-419. Available: https://doi.org/10.1016/B978-0-08-097086-8.12185-3

**[19]** „Optical character recognition - History." *ABBYY Technology*. https://www.abbyy.co.il/?categoryId=72050&itemId=168963. (Accesat 22.06.2022).

**[20]** HAUGER, J. S. Reading Machines for the Blind. *Faculty of the Virginia Polytechnic Institute and State University*. Blacksburg, Virginia, April 1995, pp. 11-13.

**[21]** „Ground truth." *Wikipedia*, Wikimedia Foundation, 12 June 2022, https://en.wikipedia.org/wiki/Ground_truth (Accesat 2.04.2023).

**[22]** LEMOIGNE, Y., CANER, A. Molecular Imaging: Computer Reconstruction and Practice. 2016.

**[23]** DOERMANN, D. S., TOMBRE, K., Handbook of Document Image Processing and Recognition. *Springer*, eds. 2014.

**[24]** PIOTROWSKI, M. Natural Language Processing for Historical Texts. *Morgan & Claypool Publishers*, 2012.

**[25]** CAROLYN, S. MCNAMARA, D., WODAK, J., WOOD, I. "Mining for the Meanings of a Murder: The Impact of OCR Quality on the Use of Digitized Historical Newspapers." *Digital Humanities Quarterly* 8 (1). 2014 http://www.digitalhumanities.org/dhq/vol/8/1/000168/000168.html.

**[26]** UWE, S., NAJOCK, D., MORGENROTH, H., SCHMID, H., GOTSCHAREK, A., FINK, F. "OCR of historical printings of Latin texts: problems, prospects, progress." In: *Proceedings of the First International Conference on Digital Access to Textual Cultural Heritage*, 2014, pp. 57–61. DATeCH '14. New York, NY, USA: ACM. doi:10.1145/2595188.2595197.

**[27]** „ABBYY FineReader: Powered by AI.", https://pdf.abbyy.com/blog/finereader-powered-by-ai/ (Accesat 02.04.2023).

**[28]** SPRINGMANN, U., LÜDELING, A. OCR of historical printings with an application to building diachronic corpora: a case study using the RIDGES herbal corpus. *arXiv preprint* arXiv:1608.02153 (2016)

**[29]** **BUMBU, T.**, COJOCARU, S., COLESNICOV, A., MALAHOV, L., UNGUR, S. User Interface to Access Old Romanian Documents. In: *Proceedings of the 4th Conference of Mathematical Society of Moldova CMSM4-2017*, June 25-July 2, 2017, 479–482.

**[30]** **BUMBU, T**. Towards a Font Classification Model for Romanian Cyrillic Documents. *Computer Science Journal of Moldova*, v.29, n.3 (87), 2021, pp.291-298.

**[31]** BREUEL, T. M., ADNAN, UL-H., MAYCE, A. AL-A., FAISAL, S. "High-Performance OCR for Printed English and Fraktur Using LSTM Networks." In: *2th International Conference on Document Analysis and Recognition (ICDAR)*, 2013, pp. 683–687. IEEE.

**[32]** „The RIDGES project", http://korpling.german.hu-berlin.de/ridges/index_en.html (Accesat 02.04.2023).

**[33]** UWE, S., FINK, F., SCHULZ, KLAUS-U. "Automatic quality evaluation and (semi-) automatic improvement of mixed models for OCR on historical documents.", 2016, Available: http://arxiv.org/abs/1606.05157.

**[34]** „Tesseract OCR project", https://github.com/tesseract-ocr (Accesat 7.06.2022).

**[35]** SHAMS, S. Breaking down Tesseract OCR. In: *Machine Learning Medium*, 2019. Available: https://machinelearningmedium.com/2019/01/15/breaking-down-tesseract-ocr.

**[36]** DUDCZAK, A., NOWAK, A., PARKOŁA, T. "Creation of Custom Recognition Profiles for Historical Documents." In: *Proceedings of the First International Conference on Digital Access to Textual Cultural Heritage*, 2014, pp. 143–146. ACM.

**[37]** „Eighteenth Century Collections Online", http://quod.lib.umich.edu/e/ecco/ (Accesat 13.06.2022).

**[38]** TAYLOR, BERG-K., KLEIN, D. Improved Typesetting Models for Historical OCR. In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics* (Volume 2: Short Papers), 2014, pp. 118–123. Baltimore, Maryland: Association for Computational Linguistics. Available: http://www.aclweb.org/anthology/P14-2020.

**[39]** HELIŃSKI, M., KMIECIAK, M., PARKOŁA, T. Report on the comparison of Tesseract and ABBYY FineReader OCR engines. *PCSS*, 2012, 24 p.

**[40]** „ABBYY: Recognition with Pattern Training", https://guides.nyu.edu/abbyy/training-abbyy (Accesat 10.06.2022).

**[41]** WHITE, N. Training Tesseract for Ancient Greek OCR. *Eutypon* (28–29), 2013, pp. 1–11.

**[42]** NAYAK, M., NAYAK, A.K. Odia characters recognition by training Tesseract OCR engine. In: *IJCA Proceedings on International Conference on Distributed Computing and Internet Technology 2014 ICDCIT-2014*, 2014, pp 25-30.

**[43]** CLAUSNER, C., ANTONACOPOULOS, A., PLETSCHACHER, S. Efficient and effective OCR engine training. *Int. J. Document Anal. Recognit*. 23(1), 2020, pp. 73-88.

**[44]** CLAUSNER, C., PLETSCHACHER, S., ANTONACOPOULOS, A. Aletheia— an advanced document layout and text ground-truthing system for production environments. In: *Proceedings of the 11th International Conference on Document Analysis and Recognition (ICDAR2011)*, Beijing, China, 2011, pp. 48–52.

**[45]** PLETSCHACHER, S., ANTONACOPOULOS, A. The PAGE (page analysis and ground-truth elements) format framework. In: *Proceedings of the 20th International Conference on Pattern Recognition* (ICPR2010), Istanbul, Turkey, IEEE-CS Press, 2010, pp. 257–260.

**[46]** „IMPACT Centre of Competence for Digitisation in Europe", http://www.digitisation.eu/ (Accesat 12.06.2022).

**[47]** „PRImA Text Evaluation tool", http://www.primaresearch.org/tools/PerformanceEvaluation (Accesat 12.06.2022).

**[48]** RICE, S.V. Measuring the accuracy of page-reading systems. *Ph.D. Thesis*, University of Nevada, Las Vegas, 1996, 81 p.

**[49]** SPRINGMANN, U., NAJOCK, D., MORGENROTH, H., SCHMID, H., GOTSCHAREK, A., FINK, F. OCR of historical printings of latin texts: problems, prospects, progress. In: *Proceedings of the First International Conference on Digital Access to Textual Cultural Heritage*, 2014, pp. 71–75.

**[50]** SHAFAIT, F. Document image analysis with OCRopus. In: *Multi-topic Conference*, 2009. INMIC 2009. IEEE 13th International, 2009, pp. 1-6.

**[51]** WICK, C., REUL, C., PUPPE, F. Calamari—a high-performance TensorFlow-based deep learning package for optical character recognition. *arXiv preprint* arXiv:1807.02004 (2018)

**[52]** VALUEVA, M.V., NAGORNOV, N.N., LYAKHOV, P.A., VALUEV, G.V., CHERVYAKOV, N.I. Application of the residue number system to reduce hardware costs of the convolutional neural network implementation. *Mathematics and Computers in Simulation. Elsevier BV,* 2020, pp. 232–243.

**[53]** DROBAC, S., LINDÉN, K. "Optical character recognition with neural networks and post-correction with finite state methods." *International Journal on Document Analysis and Recognition* (IJDAR), 2020, pp. 1 – 17.

**[54]** WICK, C., REUL, C., PUPPE, F. Comparison of OCR accuracy on early printed books using the open source engines Calamari and OCRopus. *JLCL 33*, 2018, pp. 79–96.

**[55]** „Digital Materials of Finland: The newspaper collection.", https://digi.kansalliskirjasto.fi/search?formats=NEWSPAPER (Accesat 15.06.2022).

**[56]** ALLEN, D. M. The Relationship between Variable Selection and Data Augmentation and a Method for Prediction. Technometrics, 1974, 16 (1), pp. 125–127. doi:10.2307/1267500. JSTOR 1267500.

**[57]** STONE, M. Cross-Validatory Choice and Assessment of Statistical Predictions. *Journal of the Royal Statistical Society, Series B (Methodological)*, 1974, 36 (2), pp. 111–147. doi:10.1111/j.2517-6161.1974.tb00994.x.

**[58]** CHRISTENSEN, R. Thoughts on prediction and cross-validation". *Department of Mathematics and Statistics University of New Mexico*, 2015, 7 p. Available: https://math.unm.edu/~fletcher/Prediction.pdf

**[59]** KAUPPINEN, P. OCR Post-Processing by Parallel Replace Rules Implemented as Weighted Finite-State Transducers. *Master's thesis, Helsingfors universitet,* 2016, Abstract. Available: https://helda.helsinki.fi/handle/10138/162866

**[60]** DROBAC, S., KAUPPINEN, P., LINDÉN, K. Improving OCR of historical newspapers and journals published in Finland. In: *Proceedings of the 3rd International Conference on Digital Access to Textual Cultural Heritage*, 2019, pp. 97–102.

**[61]** LEVENSHTEIN, V.I. Binary codes capable of correcting deletions, insertions, and reversals. *Sov. Phys. Dokl,* 1966, Vol. 10 (8), pp. 707-710.

**[62]** „What is METS/ALTO?", https://veridiansoftware.com/knowledge-base/metsalto (Accesat 11.06.2022)

**[63]** LENC, L., MARTÍNEK, J., KRÁL, P., NICOLAOU, A., CHRISTLEIN, V. HDPA: historical document processing and analysis framework. *Evolving Systems. Evolving Systems*, 2021, pp. 177-190.

**[64]** AHMADI, E., AZIMIFAR, Z., SHAMS, M., FAMOURI, M., SHAFEE, MJ. Document image binarization using a discriminative structural classifier. *Pattern Recogn Lett*, 2015, pp. 36–42.

**[65]** SAUVOLA, J., PIETIKÄINEN, M. Adaptive document image binarization. *Pattern Recogn,* 2000, Vol 33(2), pp. 225–236.

**[66]** POSTL, W. Method for automatic correction of character skew in the acquisition of a text original in the form of digital scan results. *US Patent 4,723,297*, (1988).

**[67]** RONNEBERGER, O., FISCHER, P., BROX, T. U-net: Convolutional networks for biomedical image segmentation. In: *International conference on medical image computing and computer-assisted intervention*, Springer, 2015, pp 234–241.

**[68]** CLAUSNER, C., PAPADOPOULOS, C., PLETSCHACHER, S., ANTONACOPOULOS, A. The ENP image and ground truth dataset of historical newspapers. In: 2015 13th international conference on document analysis and recognition *(ICDAR)*, IEEE, 2015, pp 931–935.

**[69]** „Bavarian-Czech network of digital historical sources." https://www.portafontium.eu/ (Accesat 21.06.2022).

**[70]** GRÜNING, T., LEIFERT, G., STRAUß, T., MICHAEL, J., LABAHN, R. A two-stage method for text line detection in historical documents. *Int J Doc Anal Recognit*, 2019, 22(3), pp. 285-302.

**[71]** SHI, B., BAI, X., YAO, C. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE Trans Pattern Anal Mach Intell*, 2017, 39(11): pp. 2298–2304.

**[72]** BAIRD, K.S. Anatomy of a versatile page reader. *Proceedings of the IEEE*, 1992, 80 (7): pp. 1059–1065.

**[73]** CATTONI, R., COIANIZ, T., MESSELODI, S., MODENA, C. M. Geometric Layout Analysis Techniques for Document Image Understanding: a Review. *ITC-irst Technical Report*, 1998, 68 p.

**[74]** CLAUSNER, C., ANTONACOPOULOS, A., PLETSCHACHER, S. ICDAR2019 Competition on Recognition of Documents with Complex Layouts - RDCL2019, *2019 International Conference on Document Analysis and Recognition (ICDAR)*, 2019, pp.1521-1526.

**[75]** STRAUß, T., WEIDEMANN, M., MICHAEL, J., LEIFERT, G., GRÜNING, T., LABAHN, R. System Description of CITlab's Recognition & Retrieval Engine for ICDAR2017

Competition on Information Extraction in Historical Handwritten Records, *icdar2017*, 2018. Available: https://arxiv.org/abs/1804.09943.

**[76]** Proiectul „Transkribus”, https://readcoop.eu/transkribus (Accesat 25.06.2022)

**[77]** „DFG-funded Initiative for Optical Character Recognition Development”, https://ocr-d.de/ (Accesat 25.06.2022)

**[78]** REUL, C., CHRIST, D., HARTELT, A., BALBACH, N., WEHNER, M., SPRINGMANN, U., WICK, C., GRUNDIG, C., BÜTTNER, A., PUPPE, F. Ocr4all— an open-source tool providing a (semi-) automatic OCR workflow for historical printings. *arXiv preprint* arXiv:1909.04032, 2019.

**[79]** CHERNYSHOVA, Y.S., GAYER, A.V., SHESHKUS, A.V. Generation method of synthetic training data for mobile OCR system. In: *Tenth international conference on machine vision* (ICMV 2017), vol. 10696, id. 106962G. SPIE, Vienna, 2018. 10.1117/12.2310119

**[80]** MARGNER, V., PECHWITZ, M. Synthetic data for Arabic ocr system development. In: *Proceedings of the sixth international conference on document analysis and recognition*, 2001. IEEE, 2001, pp 1159–1163.

**[81]** COJOCARU, S., COLESNICOV, A., MALAHOV, L., **BUMBU, T.** Optical Character Recognition Applied to Romanian Printed Texts of the 18th-20th Century. In: Computer Science Journal of Moldova. 2016, nr. 1(70), pp. 106-117. ISSN 1561-4042

**[82]** EGER, S., VOR DER BRÜCK, T., MEHLER, A.A comparison of four character-level string-to-string translation models for (OCR) spelling error correction. *Prague Bull. Math. Ling.* 2016, pp. 77–99.

**[83]** LLOBET, R., CERDAN-NAVARRO, J.R., PEREZ-CORTES, J.C., ARLANDIS, J. OCR post-processing using weighted finite-state transducers. In: *2010 20th International Conference on Pattern Recognition*, 2010 pp. 2021–2024.

**[84]** CACHO, F., RAMON, J. Improving OCR Post Processing with Machine Learning Tools. *UNLV Theses, Dissertations, Professional Papers, and Capstones*, 2019, 184 p. Available: http://dx.doi.org/10.34917/16076262

**[85]** REUL, C., SPRINGMANN, U., WICK, C., AND PUPPE F. Improving OCR accuracy on early printed books by utilizing cross fold training and voting. In: *Proceedings of the 13th IAPR International Workshop on Document Analysis Systems* (DAS’18), 2018. IEEE, pp. 423–428.

**[86]** SILFVERBERG, M., KAUPPINEN, P., LINDÉN, K. Data-driven spelling correction using weighted finite-state methods. In: *Proceedings of the SIGFSM Workshop on Statistical NLP and Weighted Automata*, 2016, pp. 51–59.

**[87]** SUTSKEVER, I., VINYALS, O., LE, Q., V. Sequence to sequence learning with neural networks. (2014) arXiv:1409.3215

**[88]** GÉNÉREUX, M., STEMLE, E.W., LYDING, V., NICOLAS, L. Correcting OCR errors for German in Fraktur font. In: *Proceedings of the First Italian Conference on Computational Linguistics* (CLiC-It 2014) (2014)

**[89]** KISSOS, I., DERSHOWITZ, N. OCR error correction using character correction and feature-based word classification. In: *2016 12th IAPR Workshop on Document Analysis Systems* (DAS), IEEE, 2016, pp. 198– 203.

**[90]** ROMERO, V., TOSELLI, A.H., VIDAL, E. Multimodal Interactive Handwritten Text Transcription. *World Scientific, Singapore*, 2012, vol. 80, pp. 63–73.

**[91]** VOBL, T., GOTSCHAREK, A., REFFLE, U., RINGLSTETTER, C., SCHULZ, K.U. Pocoto—an open source system for efficient interactive postcorrection of OCRed historical texts. In: *Proceedings of the First International Conference on Digital Access to Textual Cultural Heritage*, ACM, 2014, pp. 57–61.

**[92]** SIDOROV, G., VELASQUEZ, F., STAMATATOS, E., GELBUKH, A., CHANONA-HERNÁNDEZ, L. Syntactic Dependency-Based N-grams as Classification Features. *Advances in Computational Intelligence. Lecture Notes in Computer Science*, 2012, Vol. 7630. pp. 1–11. doi:10.1007/978-3-642-37798-3_1

**[93]** ENGLMEIER, T., FINK, F., SCHULZ, K.U. AI-PoCoTo—combining automated and interactive OCR postcorrection. In: *Proceedings of the Third International Conference on Digital Access to Textual Cultural Heritage*, ACM, 2019, pp.19-24.

**[94]** HÄMÄLÄINEN, M., HENGCHEN, S. From the Past to the Future: a fully automatic NMT and word embeddings method for OCR post-correction. In: *Recent Advances in Natural Language Processing*, INCOMA, 2019, pp. 432–437.

**[95]** REYNAERT, M. Ocr post-correction evaluation of early Dutch books online-revisited. In: *Proceedings of the tenth International Conference on Language Resources and Evaluation* LREC, 2016, pp. 967–974.

**[96]** MIKOLOV, T., CHEN, K., CORRADO, G., DEAN, J. Efficient estimation of word representations in vector space. *arXiv preprint* arXiv:1301.3781, 2013.

**[97]** PĂDURARIU C., CRISTEA, D. Solution for scanned documents segmentation and letter recognition. In: *Proceedings of the 14th edition of the International Conference on Linguistic Resources and Tools for Natural Language Processing – ConsILR-2019, Ed. Universității "Alexandru Ioan Cuza" din Iași*, 2019 p. 127-137.

**[98]** CRISTEA, D., PĂDURARIU, C., REBEJA, P., ONOFREI, M. From Scan to Text. Methodology, Solutions, and Perspectives of Deciphering Old Cyrillic Romanian Documents into the Latin Script. In: *Knowledge, Language, Models*, Bulgaria, 2020, pp. 38–56.

**[99]** CRISTEA, D., REBEJA, P., PĂDURARIU, C., ONOFREI, M., SCUTELNICU, A. Data Structure and Acquisition in DeLORo – a Technology for Deciphering Old Cyrillic-Romanian Documents. In: *Proceedings of the the 16th International Conference "Linguistic Resources and Tools for Processing The Romanian Language"*, ONLINE, 13-14 December 2021, ISSN 1843-911X, pp. 59-74.

**[100]** MIRON, P. (1986 - 2008), ANDRIESCU, AL. (1986-1990, 2000-2008), ARVINTE, V., CAPROȘU, I. (1986/7-1997), MUNTEANU, E. (2010-2015), HAJA, G. (2006-2008). Monumenta linguae Dacoromanorum. Biblia 1688, Universitatea „Alexandru Ioan Cuza" Iași, Albert-Ludwigs Universität Freiburg, *Editura Universității „Alexandru Ioan Cuza"*, Iași, 1986-2015.

**[101]** MĂRĂNDUC, C., PEREZ, C. A. A Romanian dependency treebank. In: *The International Journal of Computational Linguistics and Applications*, 2015, vol. 6(2), pp. 25-40.

**[102]** HE, K., ZHANG, X., REN, S., SUN, J. Deep residual learning for image recognition, *IEEE Conference on Computer Vision and Pattern Recognition* (CVPR), 2016, pp. 770–778.

**[103]** IONESCU R.T., POPESCU M., CAHILL A. String kernels for native language identification: Insights from behind the curtains. In: *Computational Linguistics*, 2016, vol. 42(3), pp. 491-525.

**[104]** ARIAS-CASTRO, E., CHEN, G., LERMAN, G. Spectral clustering based on local linear approximations., *Electronic Journal of Statistics*, 2011, pp. 1537–1587, arXiv:1001.1323, doi:10.1214/11-ejs651, S2CID 88518155.

**[105]** GĂMAN, M., GHADAMIYAN, L., IONESCU, R.T. AND POPESCU, M.C. Self-paced learning to improve text row detection in historical documents with missing labels. *ArXiv*, abs/2201.12216. (2022)

**[106]** JIANG, L., MENG, D., ZHAO, Q., SHAN, S., HAUPTMANN, A. Self-paced curriculum learning. In: *Proceedings of AAAI*, 2015, pp. 2694–2700.

**[107]** KUMAR, M.P., PACKER, B., KOLLER, D. Self-Paced Learning for Latent Variable Models. In: *Proceedings of NIPS*, 2010, vol. 23, pp. 1189–1197.

**[108]** LIN, W., GAO, J., WANG, Q., LI, X. Pixel-Level Self-Paced Learning For Super-Resolution. In: *Proceedings of ICASSP*, 2020, pp. 2538–2542.

**[109]** DIEM, M., KLEBER, F., FIEL, S., GRÜNING, T., GATOS, B. cBAD: ICDAR2017 Competition on Baseline Detection. In: *Proceedings of ICDAR*, 2017, pp. 1355–1360.

**[110]** BOCHKOVSKIY, A. Yolov4: Optimal Speed and Accuracy of Object Detection. *ArXiv*, arXiv:2004.10934 (2020)

**[111]** GÎFU, D. The Chronology of Old Romanian Words. Globalization and National Identity. *Studies on the Strategies of Intercultural Dialogue*, 2016, vol. 3, Arhipelag XXI, Târgu-Mureș, pp. 246-262.

**[112]** CRISTEA, D., HAJA, G., FLORESCU, C., ALDEA, B., CUZA, I., PHILIPPIDE, A. THE DIGITAL FORM OF THE THESAURUS DICTIONARY OF THE ROMANIAN LANGUAGE, 2007, 12 p. Available: https://profs.info.uaic.ro/~dcristea/papers/Cristea-et-al_SPeD07.pdf

**[113]** MUNTEANU, Ș., ȚÂRA V. D. Istoria limbii române literare. Privire generală. *Editura didactică și pedagogică*, București, 1978, p. 254-256.

**[114]** BOIAN, E., CIUBOTARU, C., COJOCARU, S., COLESNICOV, A., MALAHOV, L. Digitizarea, recunoaşterea şi conservarea patrimoniului cultural-istoric. *Revista Akademos*, nr. 1 (32), 2014, pp.61-68.

**[115]** CERETEU, I. Cartea Românească Veche în Basarabia: Istorie, Circulație, Valoare Documentară. *Editura Academiei Române*, București, 2019, p. 25-47, 81-150.

**[116]** Valori Bibliofile, Rev. *Gazeta bibliotecarului*, Iunie-Iulie 2008, nr. 6-7, p. 1.

**[117]** LU, S. J., TAN, C. L. Binarization of Badly Illuminated Document Images through Shading Estimation and Compensation. *Ninth International Conference on Document Analysis and Recognition*, 2007, pp. 312-316, doi: 10.1109/ICDAR.2007.4378723.

**[118]** HELIŃSKI, M., KMIECIAK, M., PARKOLA, T. Report on the comparison of Tesseract and ABBYY FineReader OCR engines. *IMPACT Project Report*, 2012, 13 p. https://www.digitisation.eu/fileadmin/Tool_Training_Materials/Abbyy/PSNC_Tesseract-FineReader-report.pdf

**[119] BUMBU, T.** On classification of 17th century fonts using neural networks. In: *Mathematics and IT: Research and Education*. 1-3 iulie 2021, Chişinău. Chișinău, Republica Moldova: 2021, pp. 95-96.

**[120]** DESA, I., MORĂRESCU, D., PATRICHE, I., RALIADE, A., SULICĂ, I. Publicațiile periodice românești (ziare, gazete, reviste). Vol. III: Catalog alfabetic 1919–1924, pp. 235–236, 264, 368, 374, 575, 708, 1024. Bucharest: *Editura Academiei*, 1987

**[121]** COJOCARU, S.; BURTSEVA, L.; CIUBOTARU, C.; COLESNICOV, A.; DEMIDOVA, V.; MALAHOV, L.; PETIC, M.; BUMBU, T.; UNGUR, S. On Technology for Digitization of Romanian Historical Heritage Printed in the Cyrillic Script. In: Conference on Mathematical

Foundations of Informatics. 25-30 iulie 2016, Chişinău. Chişinău, Republica Moldova: "VALINEX" SRL, 2016, pp. 160-176.

**[122]** BOROȘ, T., ZAFIU, A. Transliterare automată din engleză în română. Aplicaţii şi rezultate. *Romanian Journal of Human - Computer Interaction*, 2012, Vol. 5, pp. 1-14.

**[123]** ZHANG, M. HAIZHOU, L. JIAN, Direct Orthographical Mapping for Machine Transliteration. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, 2004, pp. 716–722.

**[124]** VINTILĂ-RĂDULESCU, I. Dicționar normativ al limbii române ortografic, ortoepic, morfologic și practic, *Editura Corint*, București, 2009, p. 817.

**[125]** „LEGEA Nr. 3462 din 31.08.1989 cu privirea la revenirea limbii moldovenești la grafia latină". Parlamentul Republicii Moldova. Publicat pe 31.08.1989 în *B.Of. Nr. 009*. https://mariusmioc.wordpress.com/2009/08/30/31-august-1989/ (Accesat 4.09.2022)

**[126]** „Transliterare", *Wikipedia,* 11 May 2021, https://en.wikipedia.org/wiki/Transliterare (Accesat 02.04.2023)

**[127] BUMBU, T.** Building a Diachronic Parallel Corpus for the Alignment of the Old Romanian Texts. In: *Proceedings of the of the Conference on Mathematical Foundations of Informatics MFOI-2019,* July 3-6, 2019, Iasi, Romania, pp. 263-269.

**[128] BUMBU, T.** Evaluarea Corpusului Diacronic Paralel cu Texte Românești din Noul Testament din 1648 & 1990. În materialele conferinței științifice a doctoranzilor *„Tendințe contemporane ale dezvoltării științei: viziuni ale tinerilor cercetători"*, ediția a 9-a, vol., 10 iunie 2020, Chișinău, pp.6-12.

**[129] BUMBU, T.** On Alignment of Textual Elements in a Parallel Diachronic Corpus. In: *Computer Science Journal of Moldova*. 2020, nr. 3(84), pp. 241-248. ISSN 1561-4042.

**[130]** DRUGUS, I., **BUMBU, T.**, BOBICEV, V., DIDIC, V., BURDUJA, A., PETRACHI, A., ALEXEI, V. Punctilog: A New Method of Sentence Structure Representation. In: *Proceedings of the Workshop on Intelligent Information Systems WIIS2021*, October 14-15, 2021, Chisinau, Republic of Moldova, pp. 118-129.

**[131]** BOBICEV, V., **BUMBU, T.**, DIDIC, V., PRIJILEVSCHI, D., MORARI, G. Punctilog Compared to Dependency Grammar and Constituency Grammar. In: *Proceedings of Symposium on Logic and Artificial Intelligence SLAI2022*, January 12-16, 2022, Louisiana, USA.

**[132]** PAPINENI, K., ROUKOS, S., WARD, T., ZHU, W. J. Bleu: a method for automatic evaluation of machine translation. *ACL-2002: 40th Annual meeting of the Association for Computational Linguistics*, 2022, pp. 311–318.

**[133]** COJOCARU, S., COLESNICOV, A., MALAHOV, L., **BUMBU, T.**, UNGUR, Ș. On Digitization of Romanian Cyrillic Printings of the 17th-18th Centuries. CSJM, vol.25, no.2 (74), 2017, pp.217-225.

**[134] BUMBU, T.**, BURTSEVA, L., COJOCARU, S., COLESNICOV, A., MALAHOV, L. Platform for Digitization of Heterogeneous Documents. In: *Conference on Applied and Industrial Mathematics CAIM 2022*. Ediția a 29 (R), 25-27 august 2022, Chișinău. Chișinău, Republica Moldova: Bons Offices, 2022, pp. 170-171. ISBN 978-9975-81-074-6.

**[135]** COLESNICOV, A., MALAHOV, L., COJOCARU, S., BURTSEVA, L., **BUMBU, T.** Development of a platform for heterogeneous document recognition using convergent technology. In: *Workshop on Intelligent Information Systems. 6-8 octombrie 2022*, Chisinau. Chişinău: Valnex, 2022, pp. 104-107. ISBN 978-9975-68-461-3.

**[136]** TOMASI C., MANDUCHI, R. Bilateral filtering for gray and color images. *Sixth International Conference on Computer Vision* (IEEE Cat. No.98CH36271), 1998, pp. 839-846, doi: 10.1109/ICCV.1998.710815.

**[137] BUMBU, T.**, CAFTANATOV, O., MALAHOV, L. Revitalization of the RM Folkloric Texts from the Second Half of the 20th Century and their Diachronic Analysis. *ROMAI J.*, v.14, no.2 (2018), pp. 33–40.

**[138]** CIUBOTARU, C., DEMIDOVA, V., **BUMBU, T.** Generation of the Romanian Cyrillic lexicon for the period 1967 – 1989. In: *Proceedings IMCS-55 The Fifth Conference of Mathematical Society of the Republic of Moldova*. Chișinău. Chișinău, Republica Moldova: Tipografia Valinex, 2019, pp. 309-316. ISBN 978-9975-68-378-4.

**[139]** CIUBOTARU, C. Backtracking algorithm for lexicon generation. Computer Science Journal of Moldova, v.29, n.1 (85), 2021, pp.136-152.

**[140]** CAFTANATOV, O., **BUMBU**, **T.**, ERHAN, L., CERNEI, I., IAMANDI, V., LUPAN, V., CAGANOVSCHI, D., CURMEI, M. Discover the Moldovan Cultural Heritage through e-Moldova Portal by Using Crowdsourcing Concept. In: *Proceedings of the Workshop on Intelligent Information Systems WIIS2021*, October 14-15, 2021, Chisinau, Republic of Moldova, pp. 65-75.

# DECLARAŢIA PRIVIND ASUMAREA RĂSPUNDERII

Subsemnatul, Bumbu Tudor, declar pe răspundere personală că materialele prezentate în teza de doctorat sunt rezultatul propriilor cercetări și realizări științifice. Conștientizez că, în caz contrar, urmează să suport consecințele în conformitate cu legislația în vigoare.

Bumbu Tudor

Semnătura _____

Data 10.04.2023

# CURRICULUM VITAE

| Nume | Prenume | Funcție |
|---|---|---|
| **Bumbu** | **Tudor** | **cercetător ştiinţific stagiar** |

| Instituție | Adresă | Țară |
|---|---|---|
| **Institutul de Matematică şi Informatică "Vladimir Andrunachievici" al USM** | **MD 2028, Chişinău, str. Academiei 5** | **Republica Moldova** |

| Telefon | Email | Data nașterii |
|---|---|---|
| **+373 60 39 21 74** | **tudor.bumbu@math.md** | **11.11.1992** |

## Educație

*diplome, universități și perioade*

Studii doctorale Școala Doctorală *Științe Fizice, Matematice, ale Informației și Inginerești*, specialitatea 121.03 Programarea calculatoarelor, Universitatea de Stat din Moldova, Chișinău, 2018-prezent

Master în Științe Exacte, Universitatea Tehnică a Moldovei, Chișinău, 2015-2017

Licențiat în Științe Exacte, Universitatea Tehnică a Moldovei, Chișinău, 2012-2015

## Experiență profesională

*angajatori, posturi și perioadă*

Institutul de Matematică și Informatică "Vladimir Andrunachievici" al USM, Chișinău, cercetător științific stagiar, 2017-prezent

Universitatea Tehnică a Moldovei, Chișinău, lect. universitar, 2019-prezent

Est Computer, programator, 2016-prezent

## Cercetare

*scurtă descriere a cercetărilor curente și a domeniilor de specializare*

Domeniile mele de cercetare sunt învățarea automată și prelucrarea limbajului natural. În proiectul meu de doctorat, sunt preocupat de cercetarea și dezvoltarea tehnologiilor și resurselor informaționale pentru digitizarea și prelucrarea documentelor din patrimoniul chirilic românesc.

## Publicații

*lista publicațiilor*


1. BUMBU, T., BURTSEVA, L., COJOCARU, S., COLESNICOV, A., MALAHOV, L. A Platform for Processing Heterogeneous Documents. In: Proceedings of the the 17th International Conference "Linguistic Resources and Tools for Processing The Romanian Language", 10-12 November 2022, ISSN 1843-911X, pp. 141-151.
2. COLESNICOV, A., MALAHOV, L., COJOCARU, S., BURTSEVA, L., BUMBU, T. Development of a platform for heterogeneous document recognition using convergent technology. In: Workshop on Intelligent Information Systems. 6-8 octombrie 2022, Chişinău: Valnex, 2022, pp. 104-107. ISBN 978-9975-68-461-3.
3. BUMBU, T., BURTSEVA, L., COJOCARU, S., COLESNICOV, A., MALAHOV, L. Platform for Digitization of Heterogeneous Documents. In: Conference on Applied and Industrial Mathematics CAIM 2022. Ediția a 29 (R), 25-27 august 2022, Chișinău. Chișinău, Republica Moldova: Bons Offices, 2022, pp. 170-171. ISBN 978-9975-81-074-6.
4. BUMBU, T., COJOCARU, S., COLESNICOV, A., MALAHOV, L., UNGUR, S. User Interface to Access Old Romanian Documents. In: Proceedings of the 4th Conference of Mathematical Society of Moldova CMSM4-2017, June 25-July 2, 2017, pp. 479–482.
5. BUMBU, T. Towards a Font Classification Model for Romanian Cyrillic Documents. Computer Science Journal of Moldova, v.29, n.3 (87), 2021, pp.291-298.
6. COJOCARU, S., COLESNICOV, A., MALAHOV, L., BUMBU, T. Optical Character Recognition Applied to Romanian Printed Texts of the 18th-20th Century. In: Computer Science Journal of Moldova. 2016, nr. 1(70), pp. 106-117. ISSN 1561-4042

7. BUMBU, T. On classification of 17th century fonts using neural networks. In: Mathematics and IT: Research and Education. 1-3 iulie 2021, Chişinău. Chișinău, Republica Moldova: 2021, pp. 95-96.
8. BUMBU, T. Building a Diachronic Parallel Corpus for the Alignment of the Old Romanian Texts. In: Proceedings of the of the Conference on Mathematical Foundations of Informatics MFOI-2019, July 3-6, 2019, Iasi, Romania, pp. 263–269.
9. BUMBU, T. Evaluarea Corpusului Diacronic Paralel cu Texte Românești din Noul Testament din 1648 & 1990. În materialele conferinței științifice a doctoranzilor „Tendințe contemporane ale dezvoltării științei: viziuni ale tinerilor cercetători", ediția a 9-a, vol., 10 iunie 2020, Chișinău, pp.6-12.
10. BUMBU, T. On Alignment of Textual Elements in a Parallel Diachronic Corpus. In: Computer Science Journal of Moldova. 2020, nr. 3(84), pp. 241-248. ISSN 1561-4042.
11. DRUGUS, I., BUMBU, T., BOBICEV, V., DIDIC, V., BURDUJA, A., PETRACHI, A., ALEXEI, V. Punctilog: A New Method of Sentence Structure Representation. In: Proceedings of the Workshop on Intelligent Information Systems WIIS2021, October 14-15, 2021, Chisinau, Republic of Moldova. pp. 118-129.
12. BOBICEV, V., BUMBU, T., DIDIC, V., PRIJILEVSCHI, D., MORARI, G. Punctilog Compared to Dependency Grammar and Constituency Grammar. In: Logic and Artificial Intelligence, Chisinau, 2023, pp. 92-106.
13. COJOCARU, S., COLESNICOV, A., MALAHOV, L., BUMBU, T., UNGUR, Ș. On Digitization of Romanian Cyrillic Printings of the 17th-18th Centuries. CSJM, vol.25, no.2 (74), 2017, pp.217-225.
14. BUMBU, T., CAFTANATOV, O., MALAHOV, L. Revitalization of the RM Folkloric Texts from the Second Half of the 20th Century and their Diachronic Analysis. ROMAI J., v.14, no.2 (2018), pp. 33–40.
15. CIUBOTARU, C., DEMIDOVA, V., BUMBU, T. Generation of the Romanian Cyrillic lexicon for the period 1967 – 1989. In: Proceedings IMCS-55 The Fifth Conference of Mathematical Society of the Republic of Moldova. Chișinău. Chișinău, Republica Moldova: Tipografia Valinex, 2019, pp. 309-316. ISBN 978-9975-68-378-4.
16. CAFTANATOV, O., BUMBU, T., ERHAN, L., CERNEI, I., IAMANDI, V., LUPAN, V., CAGANOVSCHI, D., CURMEI, M. Discover the Moldovan Cultural Heritage through e-Moldova Portal by Using Crowdsourcing Concept. In: Proceedings of the Workshop on Intelligent Information Systems WIIS2021, October 14-15, 2021, Chisinau, Republic of Moldova, pp. 65-75.
17. BOBICEV, V., BUMBU, T., LAZU, V., MAXIM, V., ISTRATI, D. Folk Poetry for Computers: Moldovan Codri's Ballads Parsing. In: PROCEEDINGS OF THE 12 TH INTERNATIONAL CONFERENCE "LINGUISTIC RESOURCES AND TOOLS FOR PROCESSING THE ROMANIAN LANGUAGE" MĂLINI, 27-29 OCTOBER 2016, pp. 39-50.